

University of South Wales



2059336



105 Cathays Terrace, Cardiff CF24 4HU, U.K.

Tel: +44 (0)29 2039 5882

Email: [info@bookbindersuk.com](mailto:info@bookbindersuk.com)

[www.bookbindersuk.com](http://www.bookbindersuk.com)

**Application of  
Clustering Techniques to the  
Classification of Marine  
Phytoplankton**

Samantha Ann Hardy

A submission presented in partial fulfilment of the requirements of the  
University of Glamorgan / Prifysgol Morgannwg for the degree of  
Master of Philosophy

This research programme was carried out in collaboration with Cardiff  
University

July 2004

**Application of  
Clustering Techniques to the  
Classification of Marine  
Phytoplankton**

Samantha Ann Hardy

A submission presented in partial fulfilment of the requirements of the  
University of Glamorgan / Prifysgol Morgannwg for the degree of  
Master of Philosophy

This research programme was carried out in collaboration with Cardiff  
University

July 2004

# UNIVERSITY OF GLAMORGAN PRIFYSGOL MORGANNWG

(This form should be typewritten)

## Candidate's Declaration Form

*Note: This form must be submitted to the University with the candidate's thesis (10.5 of the Regulations refers)*

Name of Candidate: Sam Hardy

Degree for which thesis is submitted: MPhil

**1. Statement of advanced studies undertaken in connection with the programme of research (Regulation 4.1 refers)**

I attended courses on Research Methodology and How to get Published, as well as various staff development programmes e.g. Creative Problem Solving.

**2. Concurrent registration for two or more academic awards (Regulation 4.8 refers)**

either\* I declare that while registered as a candidate for the University of Glamorgan's research degree, I have not been a registered candidate or enrolled student for another award of the University or other academic or professional institution.

or ~~\*I declare while registered for the University of Glamorgan's research degree, I was, with the University's specific permission, a \*registered candidate/\*enrolled student for the following award.~~

**3. Material submitted for another award**

either I declare that no material contained in the thesis has been used in any other submission for any academic award.

or ~~I declare that the following material contained in the thesis formed part of a submission for the award of~~ .....

~~(state award and awarding body and list the material below)~~

Signature of candidate: .....

S Hardy

Date: .....

23/6/04

\* delete as appropriate





## Approval for Submission of a Thesis

Student: Sam Hardy

Award: MPhil

**This is to confirm that a thesis entitled:**

Application of Clustering Techniques to  
The Classification of Marine Phytoplankton

has been approved for submission by the student's supervisory team and can therefore be forwarded to the examiners.

Signed

CW Morris  
Director of Studies

Date

23/06/04

Confirm Thesis

## **Abstract**

An introduction to classification methods is given with sections on Flow Cytometry, Clustering and so on. This includes a literature survey on research into fuzzy clustering algorithms, with a section specifically related to Flow Cytometry. Details are given about the data sets and the software used, and the clustering algorithms investigated. The flow cytometry data was collected for marine phytoplankton. Two groups of data are used, one containing species that overlapped each other, and one containing independent species (non-overlapped). Ten clusters of 1000 records each are collated for each group, each record comprising of seven variables. Six clustering algorithms (Fuzzy K-Means, Adaptive Distances, Fuzzy K-Means, Generalised Distances, Maximum Likelihood, Minimum Total Volume, and Sum of all Normalised Determinants) are used to cluster the flow cytometry data. The results are compared for each group of data, for each algorithm, based on the number of clusters produced and the relationships of the phytoplankton species placed in each cluster. Conclusions are drawn about the suitability of each algorithm to cluster phytoplankton flow cytometry data, and a discussion follows on some flow cytometry data issues. Various potential algorithms that could be investigated in future research are discussed.

## **Acknowledgements**

Many thanks go to my supervisors, Mr Colin Morris and Professor Lynne Boddy for arranging and providing the funding for this project (EU grant 3101-E264) and their help and guidance in writing this thesis and the published work that resulted from this research.

I would particularly like to mention Malcolm F. Wilkins who's supervision and input was invaluable. Thank you M.

The *AimsNet* software used was developed by Malcolm F. Wilkins as part of AIMS, a project funded by the Commission of the European Community, CEC grant no. MAS3-CT97-0080.

**(See <http://www.cf.ac.uk/biosi/staff/wilkins/aimsnet>)**

Thanks are also due to Dr. Glen Tarran and Dr Peter Burkill of Plymouth Marine Labs, for provision of the AFC data on which this work was based, collected as part of a Natural Environment Research Council PRiME Special Topic Award (GST/02/1062).

Finally, I would like to mention 'the other three witches' with whom I spent many happy hours partaking in the essential researcher activity of "socialising" and coffee drinking.

## Table of Contents

Candidate's Declaration Form .....	ii
Approval for Submission of a Thesis.....	iii
Abstract .....	iv
Acknowledgements .....	v
List of Tables.....	x
List of Figures .....	xii
List of Conferences, Posters and Publications.....	xiii
Chapter 1 - Introduction .....	1
1.1 PHYTOPLANKTON AND FLOW CYTOMETRY.....	1
<b>1.1.1 Phytoplankton</b> .....	1
<b>1.1.2 Importance of Phytoplankton</b> .....	1
<b>1.1.3 Types of Phytoplankton</b> .....	3
1.1.3.1 Characteristics of the Plankton Classes.....	4
<b>1.1.4 Conventional Data Gathering Techniques</b> .....	8
<b>1.1.5 Automated Cell Analysers: The Flow Cytometer</b> .....	8
<b>1.1.6 The EurOPA Project</b> .....	11
1.1.6.1 Deficiencies of Commercial Flow Cytometers for Marine Use.....	11
1.1.6.2 Development of EurOPA Instrument specifically for Phytoplankton Analysis .....	12
1.2 DATA ANALYSIS .....	13
<b>1.2.1 The "Data Analysis Bottleneck"</b> .....	13
<b>1.2.2 Problems Inherent in Flow Cytometric Data</b> .....	14
<b>1.2.3 Two-Dimensional Scatter Plots</b> .....	15
<b>1.2.4 Principal Components Analysis – Three Dimensional Scatter         Plots</b> .....	16
<b>1.2.5 Statistical Multivariate Analysis</b> .....	19
<b>1.2.6 Pattern Recognition</b> .....	22
1.2.6.1 Introduction to Pattern Recognition .....	22
1.2.6.2 Pattern recognition as a multivariate clustering and partitioning problem.....	23
1.2.6.3 Memorisation and Generalisation.....	24
<b>1.2.7 Bayesian Statistical Pattern Recognition</b> .....	24
<b>1.2.8 Neural Network Pattern Recognition</b> .....	25
1.2.8.1 Introduction to Neural Networks .....	25

1.2.8.2 Artificial Neurons .....	26
1.2.8.3 Artificial Neural Networks (ANNs).....	27
1.2.8.4 Training .....	28
1.2.8.5 Testing .....	29
1.2.8.6 The Expected Output .....	29
1.2.8.7 Features of an ANN.....	29
1.2.8.8 Neural Networks and Data Analysis .....	30
1.2.8.9 Applications of Neural Networks.....	30
<b>1.2.9 Clustering</b> .....	<b>32</b>
1.2.9.1 Exclusive and Non-Exclusive .....	33
1.2.9.2 Intrinsic and Extrinsic .....	33
1.2.9.3 Hierarchical and Non-Hierarchical.....	34
1.2.9.4 Hierarchical Clustering Algorithms .....	35
1.2.9.4.1 Agglomerative and Divisive .....	36
1.2.9.4.2 Serial and Simultaneous .....	36
1.2.9.4.3 Monothetic and Polythetic.....	36
1.2.9.4.4 Graph Theory and Matrix algebra .....	37
1.2.9.5 Non-Hierarchical Clustering Algorithms.....	37
1.2.9.6 Fuzzy Clustering.....	38
1.2.9.7 Clustering Algorithms and Flow Cytometry Data .....	38
1.2.9.8 Distance Metrics.....	39
1.2.9.8.1 The Euclidean Distance Metric .....	40
1.2.9.8.1.1 Correlated Features .....	40
1.2.9.8.1.2 Curved Boundaries .....	41
1.2.9.8.1.3 Complex Feature Space.....	41
1.2.9.8.2 The Mahalanobis Distance Metric.....	42

## **Chapter 2 - Literature Survey.....43**

2.1 WHY USE CLUSTERING ALGORITHMS? .....	43
2.2 FUZZY CLUSTERING ALGORITHMS AND APPLICATIONS.....	44
<b>2.2.1 From a fuzzy clustering in flow cytometry point of view</b> .....	<b>46</b>
2.3 SUMMARY OF LITERATURE SURVEY .....	47
2.4 REAL WATER SAMPLES .....	50
2.5 HOW MANY GROUPS ARE THERE REALLY IN OUR DATA? .....	51
2.6 DO WE NEED TO REDEFINE THE TAXONOMIC GROUPS?.....	52

## **Chapter 3 – Materials and Methods.....54**

3.1 DATA SETS .....	54
3.2 EXPERIMENTAL PROCEDURE.....	56
3.3 COMPUTER HARDWARE AND SOFTWARE.....	57
3.4 COMPARING THE RESULTS OF TWO CLUSTERINGS.....	60
3.5 EVALUATING ALGORITHM PERFORMANCE.....	62

**Chapter 4 – Clustering Algorithms.....63**

4.1 INTRODUCTION.....	63
4.2 FUZZY K-MEANS FKM.....	65
<b>4.2.1 Results</b> .....	68
4.2.1.1 <i>Non-Overlapping Data</i> .....	68
4.2.1.1.1 Six Clusters.....	68
4.2.1.1.2 Ten Clusters .....	68
4.2.1.1.3 Fourteen Clusters .....	69
4.2.1.2 <i>Overlapping Data</i> .....	72
4.2.1.2.1 Six Clusters.....	72
4.2.1.2.2 Ten Clusters .....	72
4.2.1.2.3 Fourteen Clusters .....	73
4.2.1.2.4 Five Clusters – One Taxon .....	74
<b>4.3.1 Results</b> .....	82
4.3.1.1 <i>Non-Overlapping Data</i> .....	82
4.3.1.1.1 Six Clusters.....	82
4.3.1.1.2 Ten Clusters .....	82
4.3.1.1.3 Fourteen Clusters .....	83
4.3.1.2.1 Six Clusters.....	87
4.3.1.2.2 Ten Clusters .....	87
4.3.1.2.3 Fourteen Clusters .....	88
4.3.1.2.4 Five Clusters – One Taxon .....	88
<b>4.4.1 Results</b> .....	96
4.4.1.1 <i>Non-Overlapping Data</i> .....	96
4.4.1.1.1 Six Clusters.....	96
4.4.1.1.2 Ten Clusters .....	96
4.4.1.1.3 Fourteen Clusters .....	97
4.4.1.2.1 Six Clusters.....	101
4.4.1.2.2 Ten Clusters .....	101
4.4.1.2.3 Fourteen Clusters .....	102
4.4.1.2.4 Five Clusters – One Taxon .....	102
<b>4.5.1 Results</b> .....	110
4.5.1.1 <i>Non-Overlapping Data</i> .....	110
4.5.1.2 <i>Overlapping Data</i> .....	110
4.5.1.2.1 Ten Clusters .....	110
4.6 SUM OF ALL NORMALISED DETERMINANTS SAND .....	112
<b>4.6.1 Results</b> .....	115
4.6.1.1 <i>Non-Overlapping Data</i> .....	115
4.6.1.1.1 Six Clusters.....	115
4.6.1.1.2 Ten Clusters .....	115
4.6.1.1.3 Fourteen Clusters .....	116
4.6.1.2.1 Six Clusters.....	120
4.6.1.2.2 Ten Clusters .....	120
4.6.1.2.3 Fourteen Clusters .....	121
<b>4.7.1 Results</b> .....	129

4.7.1.1 <i>Non-Overlapping Data</i> .....	129
4.7.1.1.1 <i>Six Clusters</i> .....	129
4.7.1.1.2 <i>Ten Clusters</i> .....	129
4.7.1.1.3 <i>Fourteen Clusters</i> .....	130
4.7.1.2.1 <i>Six Clusters</i> .....	134
4.7.1.2.2 <i>Ten Clusters</i> .....	134
4.7.1.2.3 <i>Fourteen Clusters</i> .....	135
4.7.1.2.4 <i>Five Clusters – One Taxon</i> .....	135
4.8 RESULTS SUMMARY .....	140
<b>4.8.1 Comparison Of Algorithms Using Data With Non-Overlapping Clusters</b> .....	140
4.8.1.1 <i>Six Clusters</i> .....	141
4.8.1.2 <i>Ten Clusters</i> .....	141
4.8.1.3 <i>Fourteen Clusters</i> .....	142
<b>4.8.2 Comparison Of Algorithms Using Data With Overlapping Clusters</b> .....	144
4.8.2.1 <i>Six Clusters</i> .....	145
4.8.2.2 <i>Ten Clusters</i> .....	145
4.8.2.3 <i>Fourteen Clusters</i> .....	146
4.8.2.4 <i>Five Clusters – One Taxon</i> .....	146
4.9 CONCLUSIONS .....	147
<b>Chapter 5 – Future Research .....</b>	<b>149</b>
5.1 INTRODUCTION.....	149
5.2 MINIMISING A REGULARISED COST FUNCTION ALGORITHM (RCF) .....	149
5.3 UNSUPERVISED ROBUST C MEANS ALGORITHM (URCP) .....	151
5.4 A ROBUST COMPETITIVE AGGLOMERATION ALGORITHM (RCA) .....	152
5.5 A COMPETITIVE ELLIPTICAL CLUSTERING ALGORITHM (ECL).....	153
5.6 DISCUSSION OF THE ALGORITHMS .....	154
5.7 OTHER OPTIONS .....	156
<b>Bibliography .....</b>	<b>158</b>
<b>Appendices.....</b>	<b>163</b>
APPENDIX 1 – GLOSSARY .....	163
APPENDIX 2 – WEB SITE DIRECTORY .....	164
APPENDIX 3 – CONFERENCES, POSTERS AND PUBLICATIONS .....	166

## List of Tables

Table No.		Page
1.1	Taxonomic groupings of the phytoplankton species used in this research.	4
1.2	Typical flow cytometry measurements recorded	10
3.1	List of all phytoplankton species in full data set	54
3.2	Data sets showing the species of phytoplankton used	56
4.1	Summary of times to convergence and consistency of clustering of the algorithms when using Data Set A (Non-Overlapping Clusters)	140
4.2	Summary of times to convergence and consistency of clustering of the algorithms when using Data Set B (Overlapping Clusters)	145



## List of Figures

Figure No.		Page
1.1	<i>Nodularia spumigena</i> bloom, January 2002, in the Gippsland Lakes, Victoria, Australia Photo credit: J.D. Kinnon. [Web 3]	2
1.2	<i>Prasinophyceae</i> (Planktonic Green Algae)	5
1.3	<i>Dinophyceae</i> (Dinoflagellates)	7
1.4	Schematic Diagram of A Flow Cytometer (Coulter EPICS 741) [1]	9
1.5	Two-Dimensional Scatter Plots of the Data Sets	15
1.6	Three-dimensional Scatter Plots of the Data Sets	17,18
1.7	Diagram of an Artificial Neuron	26
1.8	Diagram of a Neural Network	27
1.9	Tree of Clustering Algorithm Types	33
1.10	Example of a Dendrogram	35
3.1	AimsNet Front End	58
3.2	AimsNet Data Viewer	59
4.1	FKM Non-Overlapping Data - Six Clusters	70
4.2	FKM Non-Overlapping Data - Ten Clusters	71
4.3	FKM Overlapping Data – Six Clusters	75
4.4	FKM Overlapping Data – Ten Clusters	76
4.5	FKM Overlapping Data – Fourteen Clusters	77
4.6	FKM Overlapping Data – Five Clusters / One Taxon	78
4.7	AD Non-Overlapping – Six Clusters	84
4.8	AD Non-Overlapping – Ten Clusters	85
4.9	AD Non-Overlapping – Fourteen Clusters	86
4.10	AD Overlapping Data – Six Clusters	89
4.11	AD Overlapping Data – Ten Clusters	90
4.12	AD Overlapping Data – Fourteen Clusters	91
4.13	AD Overlapping Data – Five Clusters / One Taxon	92
4.14	ML Non-Overlapping Data – Six Clusters	98
4.15	ML Non-Overlapping Data – Ten Clusters	99
4.16	ML Non-Overlapping Data – Fourteen Clusters	100
4.17	ML Overlapping Data – Six Clusters	103
4.18	ML Overlapping Data – Ten Clusters	104
4.19	ML Overlapping Data – Fourteen Clusters	105
4.20	ML Overlapping Data – Five Clusters / One Taxon	106
4.21	MTV Overlapping Data – Ten Clusters	111
4.22	SAND Non-Overlapping Data – Six Clusters	117
4.23	SAND Non-Overlapping Data – Ten Clusters	118
4.24	SAND Non-Overlapping Data – Fourteen Clusters	119

4.25	SAND Overlapping Data – Six Clusters	123
4.26	SAND Overlapping Data – Ten Clusters	124
4.27	SAND Overlapping Data – Fourteen Clusters	125
4.28	SAND Overlapping Data – Five Clusters / One Taxon	126
4.29	DKLL Non-Overlapping Data – Six Clusters	131
4.30	DKLL Non-Overlapping Data – Ten Clusters	132
4.31	DKLL Non-Overlapping Data – Fourteen Clusters	133
4.32	DKLL Overlapping Data – Six Clusters	136
4.33	DKLL Overlapping Data – Ten Clusters	137
4.34	DKLL Overlapping Data - Fourteen Clusters	138
4.35	DKLL Overlapping Data – Five Clusters / One Taxon	139

## List of Conferences, Posters and Publications

### Conference Proceedings:

#### Poster:

Sam Hardy, Malcolm Wilkins, Lynne Boddy, Colin Morris

*Comparison of Six Clustering Algorithms to Classify Phytoplankton from Flow Cytometry (AFC) Data*

ISAC XX Conference 2000

Published in Cytometry Supplement 10 (2000)

#### Posters:

Sam Hardy, Malcolm Wilkins, Lynne Boddy, Colin Morris

*Comparison of Six Clustering Algorithms to Classify Phytoplankton from Flow Cytometric Data*

Submitted to Pont Dysgu (2001), annual doctoral seminar at University of Glamorgan.

I won a Highly Commended certificate for this poster.

### Journal Publications:

Malcolm F. Wilkins, Sam A. Hardy, Lynne Boddy, Colin W. Morris

*Comparison of Five Clustering Algorithms to Classify Phytoplankton from Flow Cytometry Data*

Cytometry 44: 210-217 (2001) [2].

## **Chapter 1 - Introduction**

### **1.1 Phytoplankton and Flow Cytometry**

#### ***1.1.1 Phytoplankton***

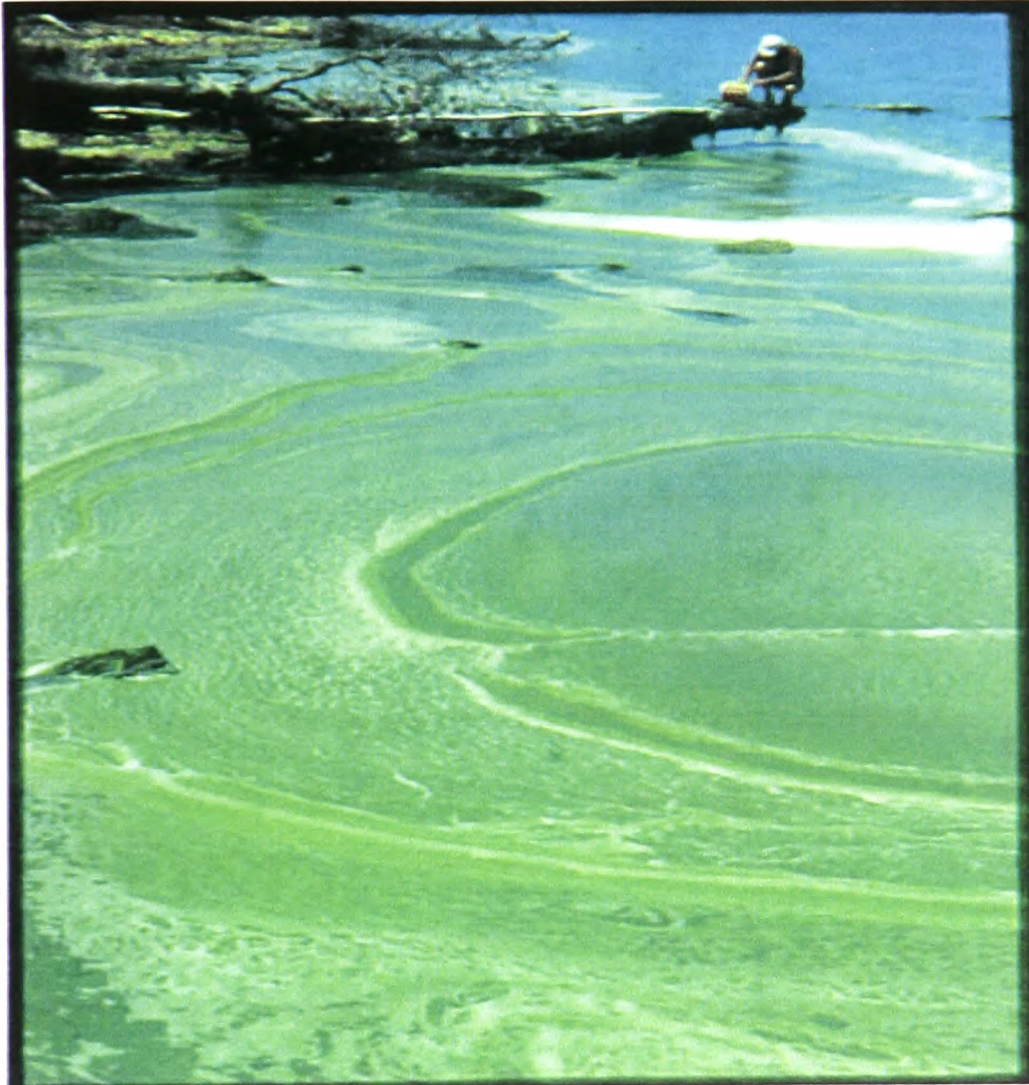
There are two main types of plankton; animals called zooplankton and plants called phytoplankton. Phytoplankton are the microscopic plants which float in both salt and fresh water, and are passively moved by wind and currents. In aquatic habitats, phytoplankton are at the bottom of the food chain, and rely on other microscopic organisms, mostly bacteria, “which convert organic material into inorganic nutrients” which they require [3].

#### ***1.1.2 Importance of Phytoplankton***

Phytoplankton have been shown to have a significant role in the way oceans affect the global climate [4]. They are responsible for the “global cycling of carbon dioxide through the fixing of around  $50 \times 10^9$  tonnes of carbon” [5] each year, and perform the fundamental function of providing “proteins, carbohydrates, fats, vitamins, and mineral salts to primary consumers” [Web 1].

While most species are harmless, pollution can sometimes cause an increase in nutrient levels in water which can in turn cause the development of large ‘blooms’ of phytoplankton, (see Figure 1.1) and in some species, toxins which can be detrimental to marine life and occasionally humans. These ‘blooms’ can block sunlight from reaching the bottom in shallow water areas and can cause the “massive decline in the submerged aquatic vegetation” [Web 2].

**Figure 1.1** *Nodularia spumigena* bloom, January 2002, in the Gippsland Lakes, Victoria, Australia Photo credit: J.D. Kinnon [Web 3].



By monitoring the population of test species, indications can be provided not only about the level of pollutants present in the waters, but also how they are dispersed by the movement of the water [6]. This illustrates how important it is to investigate changes in phytoplankton populations in order to gain an understanding of the global water systems.

### ***1.1.3 Types of Phytoplankton***

The majority of phytoplankton are single-celled organisms, however some consist of 'chains of loosely associated cells', while others form 'thread-like cell systems' or uniform celled colonial structures [3].

Phytoplankton vary enormously in size from 'ultraplankton' which are less than 5 $\mu$ , to the 'macroplankton' at more than 1mm. In this project, the species vary from 1 $\mu$  to 42 $\mu$ , and the approximate size of each group/class of phytoplankton given is based on the data collected for the 62 species contained in the database used in this project.

Phytoplankton are classified (see Table 1.1) according to several characteristics: cell shape and size; presence or not of a cell wall; chloroplasts in terms of colour, number and shape; flagella in terms of number, length, and presence or absence of bristle-like outgrowths or fine hair; and type of reserve substances stored like starch, oil and leucosin [3].

Normally phytoplankton would appear green due to the green chlorophylls found in the chloroplasts, but the presence of auxiliary pigments and the quantity of these pigments, means that phytoplankton exist in a large variety of colours. Added to this, some unicellular planktonic algae are colourless and phagocytic, i.e. they can engulf and digest solid organic particles like other plankton, instead of photosynthesising.

**Table 1.1** Taxonomic groupings of the phytoplankton species used in this research.

Group	Class	Species
Cryptophytes	<i>Cryptophyceae</i>	<i>C. appendiculata</i> , <i>H. brunnescens</i> , <i>Rhodomonas sp.</i>
Flagellates	<i>Prasinophyceae</i>	<i>M. pusilla</i> , <i>N. pyriformis</i> , <i>T. tetrathele</i>
Flagellates	<i>Rhodophyceae</i>	<i>P. pupereum</i> , <i>R. maculata</i>
Prymnesiophytes	<i>Prymnesiophyceae</i>	<i>C. chiton</i> , <i>E. huxleyi</i>
Diatoms	<i>Bacillariophyceae</i>	<i>A. coffaeiformis</i>
Dinoflagellates	<i>Dinophyceae</i>	<i>A. carterae</i> , <i>A. pigmentosum</i> , <i>G. veneficum</i> , <i>H. triquetra</i>

### 1.1.3.1 Characteristics of the Plankton Classes

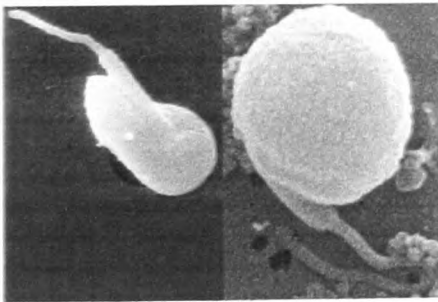
*Cryptophyceae* : There is a large variation in the chloroplast colours of this class. They range from green to brown with some species appearing blue-green or red due to the auxiliary pigments phycocyanin and phycobilin respectively. These organisms vary in size from approximately 1 $\mu$  to 25 $\mu$ . A pair of unequal length flagella “both covered with fine hair-like outgrowths” is present [3], with many species exhibiting a cell intucking or gullet to ingest organic particles.

*Cyanophyceae* (The Blue-Green Algae): This class of phytoplankton occur in unicellular, colonial and filamentous form, but are actually more similar to

bacteria than other plankton as they are prokaryotic, (i.e. they don't have a definite nucleus, nuclear membranes or chromosomes). The distinguishing characteristic of this class is that the walls of the cells consist of two or three layers close to the plasma membrane unlike other algae.

*Prasinophyceae* (Planktonic Green Algae) : These cells are approximately between 1 $\mu$  and 19 $\mu$  and mostly are green in colour. The majority are unicellular with predominantly either 2 or 4 flagella although some unflagellate organisms have been described. The flagella are thicker than other Planktonic Green Algae due to the presence of scales covering the surface of each flagellum. (See Figure 1.2)

**Figure 1.2** *Prasinophyceae* (Planktonic Green Algae)



(a) *Micromonas pusilla*



(b) *Nephroselmis pyriformis*

*Rhodophyceae* (Red Algae) : These are the family of red algae. They are red due to the presence of phycoerythrin, a pigment which reflects red light and absorbs blue. Blue light penetrates water to a deeper level than light with longer



wavelengths, enabling red algae to live and photosynthesise at a greater depth than most phytoplankton [Web11]. The life cycle of red algae often entails “three stages of independent organisms” [Web12]. Most are multicellular, where their cells are generally covered by a “slimy outer sheath”, and may contain several nuclei. Some multicellular species deposit calcium carbonate crystals both inside and outside their cell walls, and it can be difficult to differentiate these calcified red algae from corals. The cell walls may also contain “colloidal compounds”, for example, agars and carageenan. Energy from photosynthesis is stored as a carbohydrate, floridean starch. Like the pigments of red algae, this carbohydrate is unique to this class of phytoplankton [Web13].

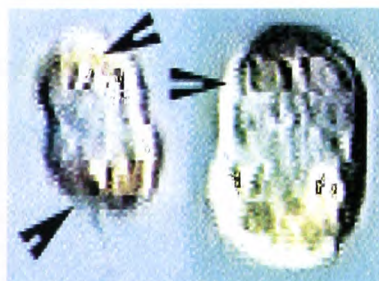
*Prymnesiophyceae* : This class of phytoplankton are in general, unicellular and photosynthetic. They may have a complex life cycle, “with an alternation between motile and non-motile phases of different morphologies” [Web 14]. Members of the group *Prymnesiophytes* were originally called *Haptophytes* due to the “presence of the unique organelle, the haptonema. This is a peg-like structure that extends out from the cell near the point where the two flagella are attached. The haptonema was originally thought to be a third flagellum, but has since been found to have a quite different morphology, and its function is unknown.” *Prymnesiophytes* occur regularly in a golden-brown colour due to the presence of diadinoxanthin and fucoxanthin pigments. Many have a covering of external plates which can either be calcified (coccoliths) or

carbohydrate based. These can have “spines or an elaborated rim”, and occur in many shapes from pentagons to donuts to trumpets [Web14].

*Bacillariophyceae* (Diatoms) : The ubiquitous Diatoms are sized between approximately  $3\mu$  and  $36\mu$  and contain xanthophyll auxiliary pigments giving them a yellow-brown colour. They possess no flagellates and are unique in having a rigid silica based cell wall.

*Dinophyceae* (Dinoflagellates) : Some of these phytoplankton occur as colourless phagocytic forms, however the majority are unicellular autotrophic organisms which obtain energy from chemical elements. The cells have an approximate size of  $7\mu$  to  $45\mu$ , some of which contain auxiliary xanthophyll pigments, resulting in a range of colours from yellow-green to yellow-brown. The cells also have two flagella, one “trailing behind the cell and lying in a groove (sulcus) and the ribbon-like transverse flagellum also lying in a groove (the cingulum or girdle)” [3]. (For examples, see Figure 1.3 below.)

**Figure 1.3** *Dinophyceae* (Dinoflagellates)



(a) *Aureodinium pigmentosum*



(b) *Heterocapsa triquetra*

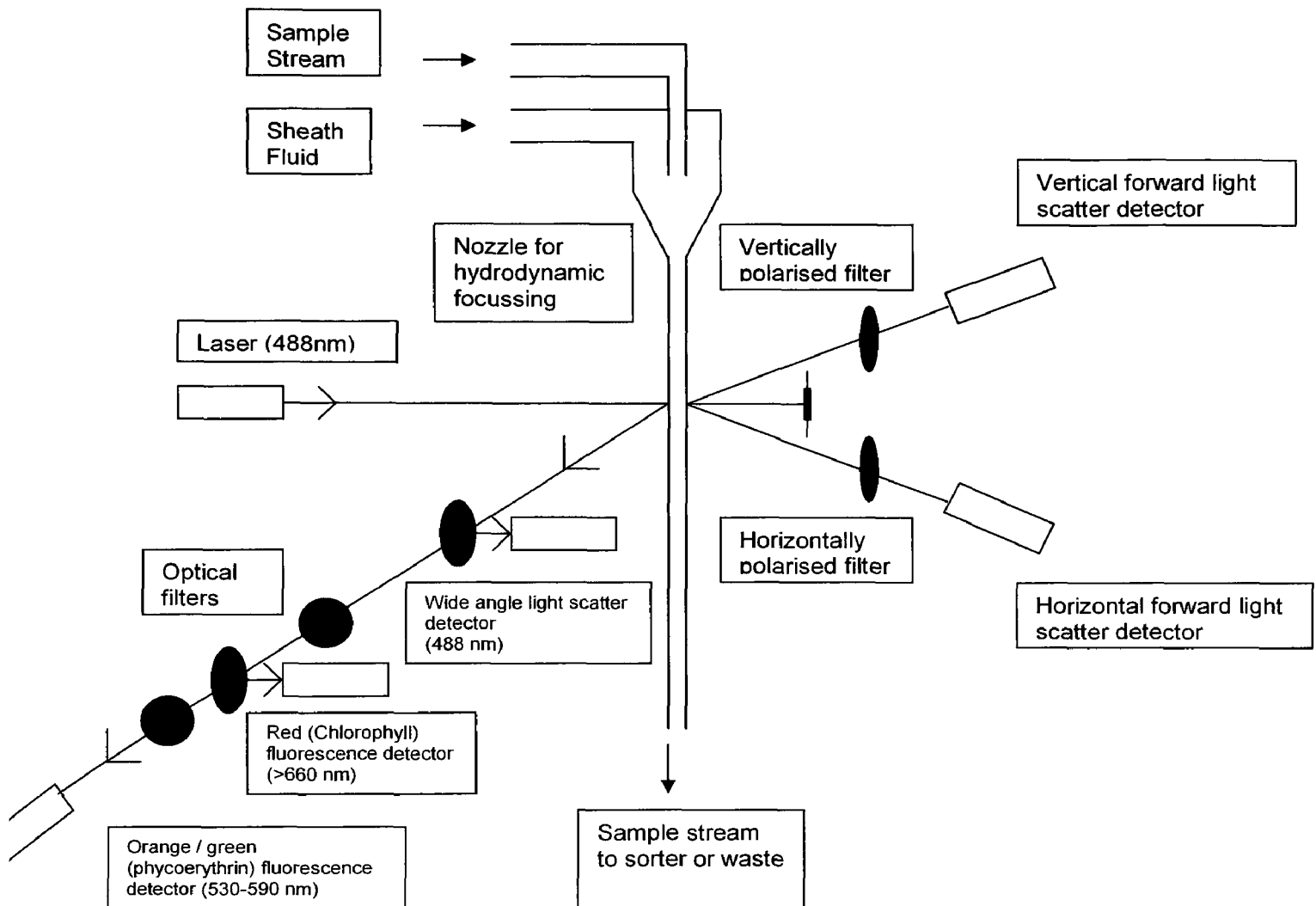
#### ***1.1.4 Conventional Data Gathering Techniques***

Established techniques for studying plankton consist firstly of sample collection, centrifuging to concentrate the cells in the sample and/or filtering. This is then followed by microscopic analysis usually involving fluorescence and staining techniques in order to highlight the presence of certain plankton [6], which the trained specialists would then be able to identify. This method is not suitable where the presence of one cell type is amongst an abundance of much larger populations of other organisms, and is extremely slow and laborious. In addition to this, as the samples are fully analysed back at the laboratory, and not in situ, if anything interesting is found, extra water samples cannot be taken for further investigation. As a result of this, “automated methods previously used to sort mammalian cells in biomedical science have been applied” [1].

#### ***1.1.5 Automated Cell Analysers: The Flow Cytometer***

Flow cytometers are sophisticated automated cell analysers, which have been applied to marine biology research [7] [8] [9]. They measure fluorescence and light scatter from single particles as cells pass singularly in suspension through a laser beam, which is used to measure properties of the particle. The cells may simply be counted, but according to the results of the measurements, they may be physically separated from the continuous stream by an “electrostatic or mechanical sorter” [1] for microscopic analysis.

**Figure 1.4** Schematic Diagram of A Flow Cytometer (Coulter EPICS 741) [1].



Modern flow cytometers consists of a light source generally in the form of a laser / lasers, collection lens, electronics and a computer to translate signals to data. Scattered and emitted fluorescent light is collected by two lenses, one set in front of the light source and one set at right angles. It gives measurements which are representative of the size and shape of the cell, and by a series of optics, beam splitters and filters, specific bands of fluorescence can be

measured. “With flushing and sample exchange, a practical analysis speed can be achieved of a few minutes per sample, with 10,000 to 100,000 particles measured” [8]. As well as this, depending on the flow cytometer used, a single cell can have between 3 and 12 variables measured as part of the analysis, compare this with the painstaking task of using microscopes to classify particles.

The properties indirectly measured by a flow cytometer include particle length, width, height and area, the presence of calcite plates covering the cell body, and the colour fluorescence of the chlorophyll. Also, internal cell complexity and, “any cell component or function that can be detected by a fluorescent compound, can be examined. (Table 1.2) This has led to the widespread use of these instruments in the biological and medical fields” [Web 4].

**Table 1.2** Typical flow cytometry measurements recorded

Name	Description
FSC-H	Log forward light scatter, - this is a measure of particle size.
SSC-H	Log side scatter - this can indicate the presence of coccoliths, calcite plates covering the cell body.
FL1 -H	Log depolarised light scatter.
FL2-H	Log peak phycoerythrin fluorescence (orange fluorescence) - used for discriminating cryptophytes and rhodophytes, which have this pigment, from species which don't.
FL3-H	Log peak chlorophyll fluorescence. This is the height of the particle.

FL3-A	Integrated chlorophyll fluorescence - this is usually closely related to measurement 5 but linear, so it doesn't provide much more information. This provides the area of the particle
FL3-W	Width of the particle.

### **1.1.6 The EurOPA Project**

#### *1.1.6.1 Deficiencies of Commercial Flow Cytometers for Marine Use*

Flow cytometry provides many advantages over optical microscopy when analysing phytoplankton. However, a number of considerations, still need to be investigated.

The principal design and use of cytometers concerns the analysis and sorting of mammalian cells. This means there is a limit to the size of particles that can be analysed, i.e. in our application area only relatively small ocean phytoplankton [8]. The optics and wavelengths of the lasers used are not conducive to analysing natural water samples. Information on the cell structure of the particle cannot be deduced using the fluorescence pulse profile. Large fragile colonies of cells tend to be disturbed by the movement of the fluid present. The number of parameters measured simultaneously may be limited by the "electronic data acquisition hardware" [1]. Most cytometers are for laboratory use only and are therefore difficult to transport.

The Optical Plankton Analyser (OPA) was designed with some of these limitations in mind; unfortunately, it was only developed as an experimental prototype and was inappropriate for fieldwork [8] [10].

### *1.1.6.2 Development of EurOPA Instrument specifically for Phytoplankton*

#### *Analysis*

The EurOPA project developed “an optical flow cytometer specialised for plankton analysis in certain areas [8] [11]:

- Three lasers, with wavelengths specifically tailored to cause fluorescence of marine phytoplankton pigments;
- Analysis of a wide range of particle sizes: the width of the laser foci (300 – 1000 $\mu$ m wide) and the low-shear fluidics are designed to allow accurate analysis of colonies as well as single cells, ranging from 1 to 1000 $\mu$ m in size;
- Improved electronics, to remove the limitation on the number of simultaneously measured parameters;
- Additional diffractometry module, to collect low angle light scatter information in much greater detail using a 5x5 detector square array: the resulting diffraction pattern contains more information on particle size and shape;
- Pulse shape analysis of light scatter and fluorescence signals, to detect chain and colony forming species;
- High speed imaging-in-flow module to capture video images of plankton;
- Purpose built sorter module to allow for the wide sample stream;
- Small size and easy portability for shipboard use, with a gimbaled suspension table to eliminate effects of ship movement;
- Integration of the entire instrument with powerful data analysis techniques” [1].

## 1.2 Data Analysis

### 1.2.1 The “Data Analysis Bottleneck”

Each time a particle in a sample passes through the interrogation point in a flow cytometer, it is classed as an *event*. Information about each event is provided by the flow cytometer - this set of measurements is known as a cytometric ‘fingerprint’ of the particle. Large numbers of such measurements can be collected rapidly, typically at a rate of more than  $10^3$  events per second with between three and twelve variables measured for each particle.

This section emphasises how much data it may be necessary to analyse in order to classify a particle when using multivariate data. Some species of phytoplankton are easily distinguishable. There may be cases where there are only two possible species that a particle could belong to which are easy to tell apart. For example, *Stichococcus bacillis* is a member of the *Cryptophyceae* class and is of a similar size to *Hemiselmis brunnescens*, also of the *Cryptophyceae* class but which fluoresces orange/green. Here just by using a threshold value on the orange/green fluorescence, the two species can be distinguished from each other. Furthermore, the threshold value could be used in flow cytometry to cause a cell sorter to collect samples of one of these species. However not all phytoplankton are so easily recognised. In which case there can be many more possible classifications of an organism, and an increase in the amount of data to be analysed may be needed. For example, when no single measurement will enable the particle to be identified as one specific species of phytoplankton, it may be required that information be used simultaneously from



all the measurements in order to decide the identity. If species from the same group are to be distinguished and not just groups of species from each other, this is particularly the case. At this stage, principal components analysis [Web 8] can be used to obtain a one or two-dimensional projection of the data with maximum variance, enabling individual clusters to be visualised. However, “in general there is no unique best projection” [1]. Flow cytometers typically have limited capabilities for analysing phytoplankton groups [12] .

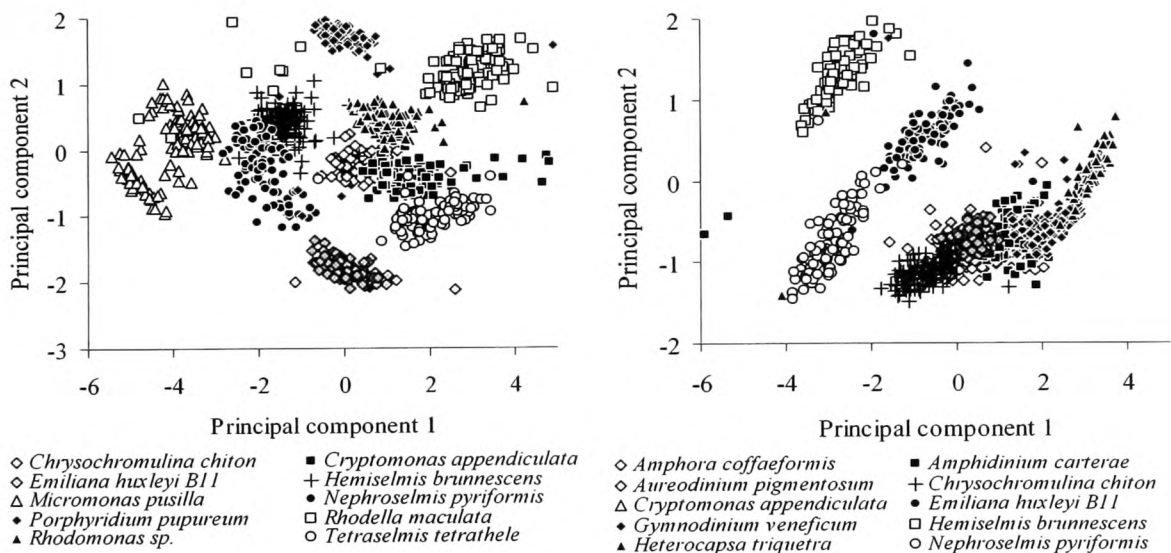
### ***1.2.2 Problems Inherent in Flow Cytometric Data***

Extensive variability is inherent in phytoplankton flow cytometry data, “even when the sample analysed has been carefully cultured under a standard abiotic regime”, [1] different strains of the same species can show variability in their cytometric ‘fingerprints’ [13]. To compound matters, cells at different stages in the species life cycle and colony formation, and cells growing under different environmental conditions will be present in the population. Samples taken from the field may also contain cells, which are eaten by other organisms. Further, there may be inorganic or mineral particles, contaminant organisms such as bacteria or other plankton species, and cell debris such as dead cells and fragments of cells, which are sample contaminants. All of these factors “can cause the data even for a nominally pure culture to appear complex” [1].

### 1.2.3 Two-Dimensional Scatter Plots

Due to the multivariate nature of the data, two-dimensional plots look very complex and very unclear. Even if different colours or shapes are used to represent the data points in the different clusters, it can still appear that all the data belongs to one big cluster, as this method does not exploit the multivariate nature of flow cytometric data. The degree of overlapping of clusters is not obvious and this method of displaying the clusters is not viable. (See Figure 1.5)

**Figure 1.5** Two-Dimensional Scatter Plots of the Data Sets



(a) Non-Overlapping Data

(b) Overlapping Data

Plots illustrating the distributions of the ten species in (a) data set A: non-overlapping data, and (b) data set B: overlapping data, projected in each case onto the plane of the first two principal components.

### **1.2.4 Principal Components Analysis – Three Dimensional Scatter**

#### **Plots**

Principal Components Analysis is a projection method used to reduce the dimensionality of data while maintaining the maximum variance.

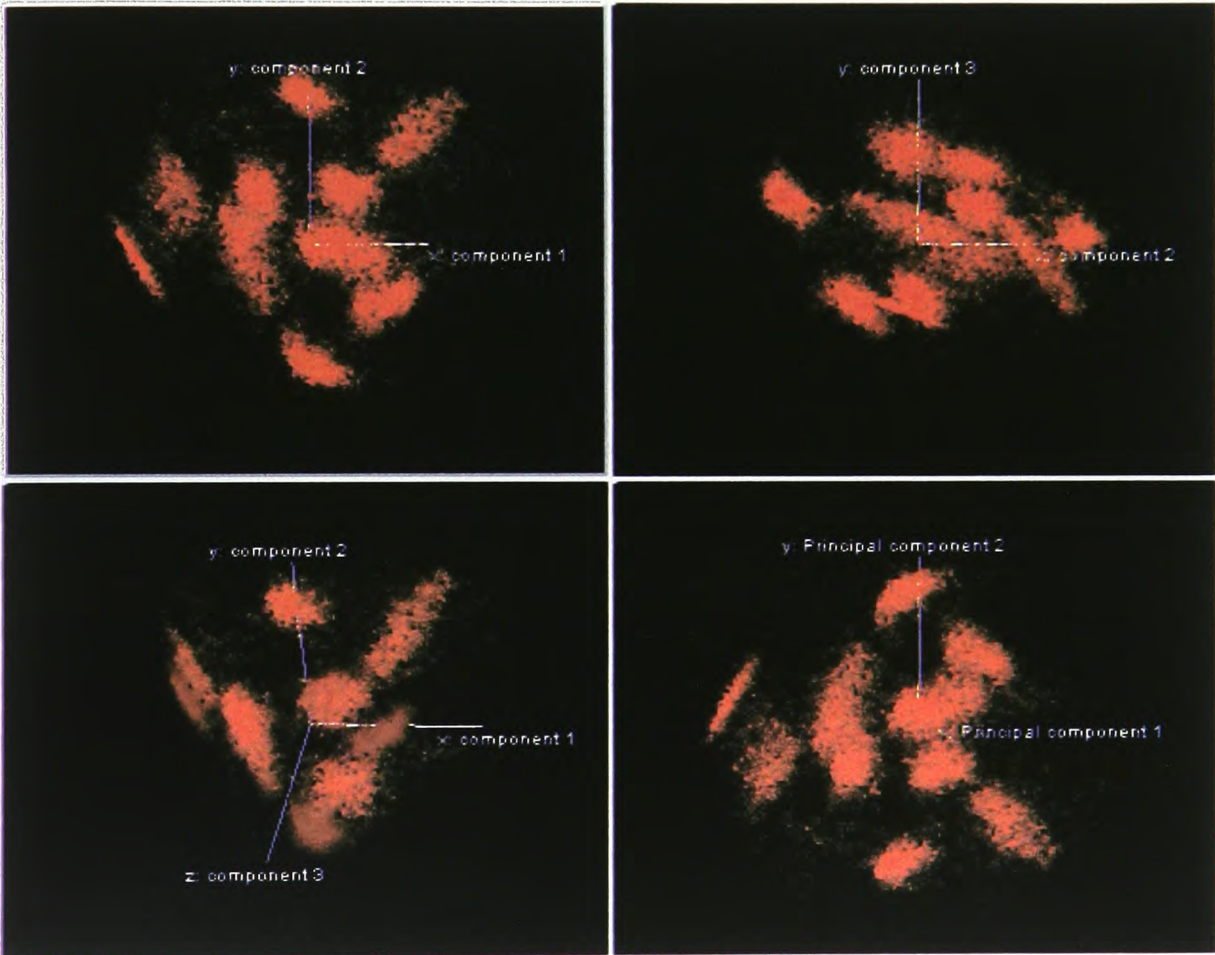
The original correlated variables are linearly transformed into independent variables called principal components. The amount of information represented by each principal component is the variance, and the components are listed in descending order of variance. Therefore, by using the data for the first  $n$  components, this method can be used to establish the  $n$ -dimensional projection of the data with maximum variance.

Typically 2-dimensional or 3-dimensional projections are applied. In this study, a 3-dimensional projection was carried out reducing the dimensionality of the data from seven to three. This meant the data could be displayed in three-dimensional space instead of the hyper-dimensional space originally required. (See Figure 1.6)

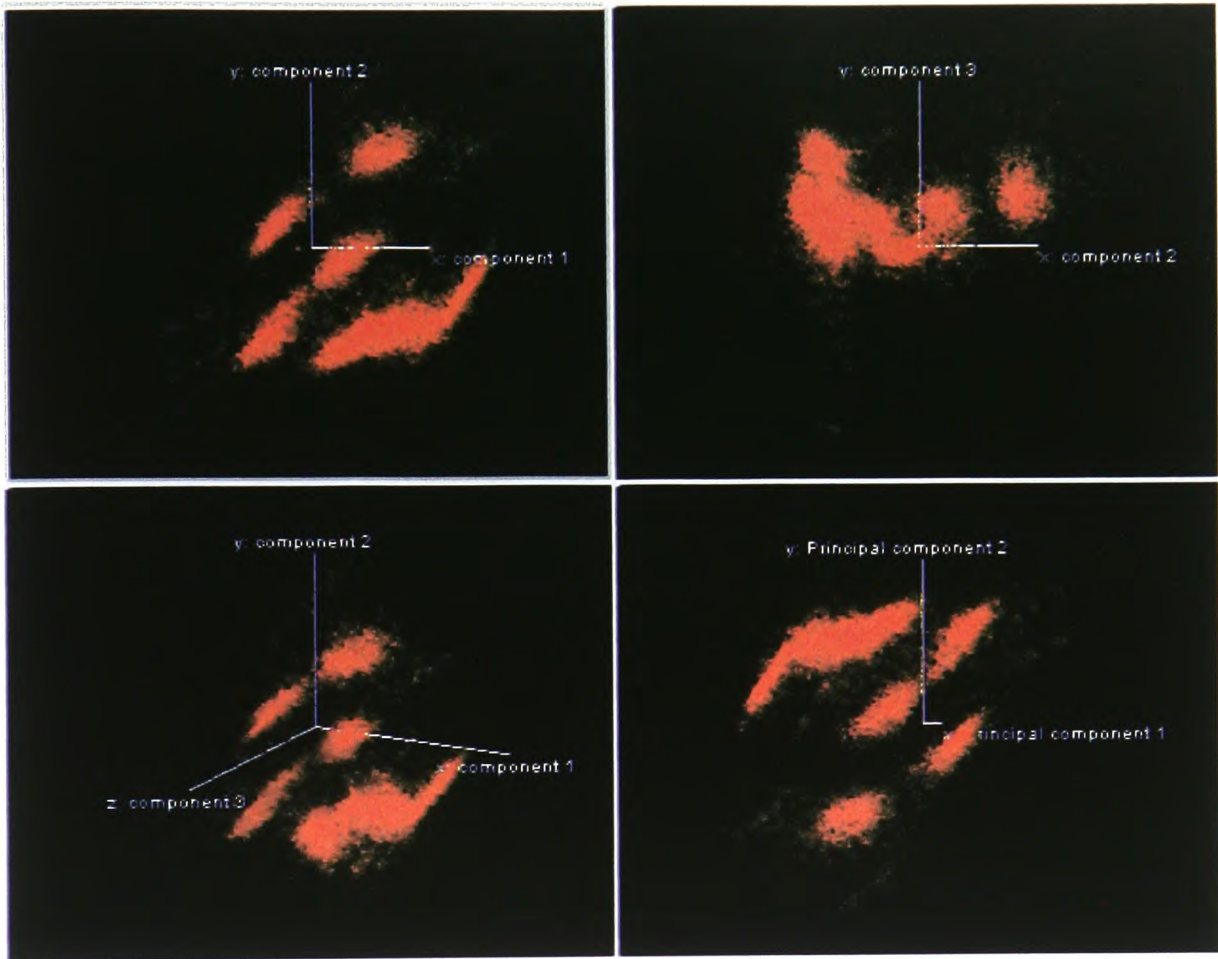
Principal component analysis can be applied to three types of square matrices, "SSCP (pure sums of squares and cross products), Covariance (scaled sums of squares and cross products), or, Correlation (sums of squares and cross products from standardised data)" [Web 8]. There is no difference in the results when using SSCP and Covariance matrices; however the Correlation type is a special case. It is necessary that Correlation matrices be used either when different units of measurement are used to record each parameter or when the variances of each parameter differ significantly. The data used here was

recorded using different measurement units, and therefore the Correlation type of Principal Component Analysis was performed.

**Figure 1.6** Three-dimensional Scatter Plots of the Data Sets



(a) Non-Overlapping Data



(b) Overlapping Data

Plots illustrating the distributions of the ten species in (a) data set A: non-overlapping data, (b) data set B: overlapping data, projected in each case onto the plane of the first three principal components.

X axis: FSC-H = Log forward light scatter (size)

Y axis: SSC-H = Log side scatter (presence of coccoliths, calcite plates covering the cell body)

Z axis: FL2-H = Log peak phycoerythrin fluorescence  
(Orange Fluorescence)

This is a useful tool to visualise the clusters, however, generally there is no single best projection relevant to all phytoplankton species. A projection allowing one pair of species to be distinguished will not be applicable for other pairs of species. Therefore, a classification approach to exploit the multivariate nature of the data is required.

### ***1.2.5 Statistical Multivariate Analysis***

A study was carried out on the use of multivariate statistical analysis to identify 32 marine phytoplankton species from their flow cytometric signatures [12]. The flow cytometric data used in the study mentioned and the data sets used in this research come from the same data source. The two methods used were Canonical Variate Analysis (CVA) and Quadratic Discrimination Analysis (QDA).

#### ***1.2.5.1 Canonical Variate Analysis (CVA)***

Canonical Correlation Analysis (CCA) looks to identify the connections present between two sets of variables as well as the strength of these connections. It is concerned with the correlation between linear combinations of the variables in each set. Firstly, the pair of linear combinations with the largest correlation is established. Secondly, the pair with the closest match of all pairs unrelated to the initially selected pair, is determined. The process continues until there are no more pairs of linear combinations to select from that are uncorrelated from the initial pair. The pairs of linear combinations are known as

the canonical variables and their correlations are referred to as the canonical correlations [14].

It has been suggested that CVA is a method similar to Principal Component Analysis in that it reduces the dimensionality of the data [Web 7].

Looking at the multivariate statistical analysis study of the 32 marine phytoplankton species [12]. Where differentiation was based on the taxonomic group, the groups that had a specific property that could be used to discriminate them were separated clearly, like Dinoflagellates which tend to be the largest species present, or entirely, like Cryptomonas which have “high orange to red fluorescence ratios” [12]. It was not possible to distinguish the other three groups. This suggests that physical cell differences between the groups “are not reflected in light scattering and fluorescence properties alone” [12].

Where species-level analysis was performed, it was found that CVA “can be a useful preliminary graphical method to apply when analysing a small number of groups” [12].

#### *1.2.5.2 Quadratic Discriminant Analysis (QDA)*

“Classificatory discriminant analysis is used to classify observations into two or more known groups on the basis of one or more quantitative variables” [Web 7]. Classification can be done by either a parametric or a non-parametric method. In cases where the distribution within each class is approximately normal, a parametric method is applicable. Methods of this type create either a linear discriminant function if the covariance matrices within each group are

assumed to be equal, or a quadratic discriminant function if the covariance matrices within each group are not equal.

In cases where the distribution within each class is assumed to have no specific distribution or a non-multivariate normal distribution, non-parametric methods can determine the criteria for classification, for example nearest-neighbour methods. “The performance of a discriminant function can be evaluated by estimating error rates [Web 7]. i.e. misclassification probabilities.

In multivariate statistical applications, the data sets analysed are generally from non-normal distributions. When a non-normal population is used to determine the classification criterion for a parametric method like QDA, the resulting error rates may be biased and therefore the evaluation of their performance could be misleading [Web 7].

In the multivariate statistical analysis of the marine phytoplankton flow cytometry data [12], QDA was found to be successful at distinguishing phytoplankton at species-level. Results showed classification rates exceeding 70% for more than two-thirds of the species, with many species well separated. However, there were low classification rates with the diatoms, but this was likely to be caused by their tendency to form chains which means they overlap with other species.

Flow cytometric signatures sometimes follow multivariate normal distributions, making QDA an appropriate method to apply as even considering the non-normal distributions; the effects of the non-normal characteristics in general, are negligible due to the clear separation of many species.



QDA proved to be a considerable improvement over “conventional AFC software” [12] in accuracy, despite a number of species being identified that were in the training data but not present in the water samples that were analysed by the flow cytometer. This was not unexpected considering that more species than were likely to be present in a single sample were included in the training data, and also because natural variation of the data caused observations to be misclassified. In addition to being successful at discriminating phytoplankton at species-level, QDA was also found to be “more than two orders of magnitude faster than conventional flow cytometric analyses for discriminating and enumerating phytoplankton species” [12].

The multivariate statistical method here compared favourably with a Back Propagation neural network approach to identify 42 marine phytoplankton strains [13]. The data sets were different; however 25 species were common to both studies.

### ***1.2.6 Pattern Recognition***

#### *1.2.6.1 Introduction to Pattern Recognition*

Pattern Recognition can be broadly described as “the interpretation of data in the presence of noise” [1], where data is matched to the required ‘object’. It has a wide range of applications: identifying stock market trends, warning of epileptic fits from EEG waves, recognising plankton from flow cytometry data, and so on.

A 'pattern' is a description of a particular instance of one of these objects (e.g. a photograph of a daffodil), and the idea of pattern recognition is to assign the instance as belonging to a particular group or class. In this example, the class could be defined as a collection of photographs containing the specific plant.

One problem to consider is that images of a particular pattern will not all look the same. A person can look very different in different images due to variation in light levels, change in background, change in profile and so on, and a pattern recognition system should be capable of allowing for this.

#### *1.2.6.2 Pattern recognition as a multivariate clustering and partitioning problem*

For convenience, data patterns are usually denoted by a list of  $n$  real numbers  $x_1, x_2, \dots, x_n$  which correspond to the values measured that constitute the pattern. It is useful to visualise these patterns as points in  $n$ -dimensional space, where each pattern is uniquely identified by a vector  $x$ , the vector from the origin to the point. Two conditions must apply in order for patterns to be identifiable: (i) Different patterns of the same object should be similar in some context. (ii) Groups of patterns of different objects must be separate from each other.

The process of pattern recognition can be considered in two parts: (i) Dividing the  $n$ -dimensional space into "labelled regions surrounding each cluster" [1]. (ii) Identifying which region a new pattern belongs to. Clusters of data points

from different groups may not be distinct, but overlap, leading to misidentification by the pattern recognition system. This is true of any such system, regardless of how successful it is.

### ***1.2.6.3 Memorisation and Generalisation***

If there was no variation in the patterns in each class, pattern recognition systems could successfully identify patterns by 'memorising' a set of training data, which would contain patterns from each known class. Each new pattern would just be compared to the examples in the training set to identify which class it belonged to.

However, due to noise contamination during data capture or where patterns in the same class are inherently variable, a pattern can differ significantly from the known classes. Here the pattern recognition system has to make a 'reasonable guess' by making 'generalisations' about the classes it knows and select the class that is the closest fit to the new pattern.

The true measure of a pattern recognition system, is the accuracy of recognition of 'unseen' test data. "Fundamentally, pattern recognition is a decision process for which the only and ultimate criterion is minimization of the average number of misclassifications" [15].

### ***1.2.7 Bayesian Statistical Pattern Recognition***

Statistical Pattern Recognition systems identify which object  $\omega_i$  an observed pattern  $x$  should be assigned to, based on the probability density

functions or likelihood functions  $p(x|\omega_i)$  for each class [16]. “This is the probability or likelihood that  $x$  could be generated by a given object  $\omega_i$ , and is related to the *a posteriori* probability  $p(\omega_i |x)$ , that  $\omega_i$  was actually the object that produced the observed  $x$ , via Bayes’ rule:

$$p(\omega_i |x) = \frac{p(x|\omega_i) p(\omega_i)}{p(x)}$$

where  $p(\omega_i)$  is the *a priori* probability of object  $\omega_i$  and  $p(x) = \sum_i p(x|\omega_i)p(\omega_i)$  is the unconditional probability of  $x$ . Where the objects are exhaustive (covering all possible classifications of  $x$ ) then  $\sum_i p(\omega_i|x) = 1$ .” The idea of a Bayes’ rule classifier, is to assign the pattern  $x$  to the object with the highest *a posteriori* probability. If all misclassifications are of equal cost, then this is known as an optimal classification strategy as the number of misidentified patterns is minimised. However, this is not always achieved, and in these cases, some misclassifications are potentially costlier than others [16] [17] [1].

## **1.2.8 Neural Network Pattern Recognition**

### **1.2.8.1 Introduction to Neural Networks**

The concept behind neural networks is that they model the operation of the human brain. To do this, they utilise a parallel processing structure that has a large number of processors and many interconnections between them.

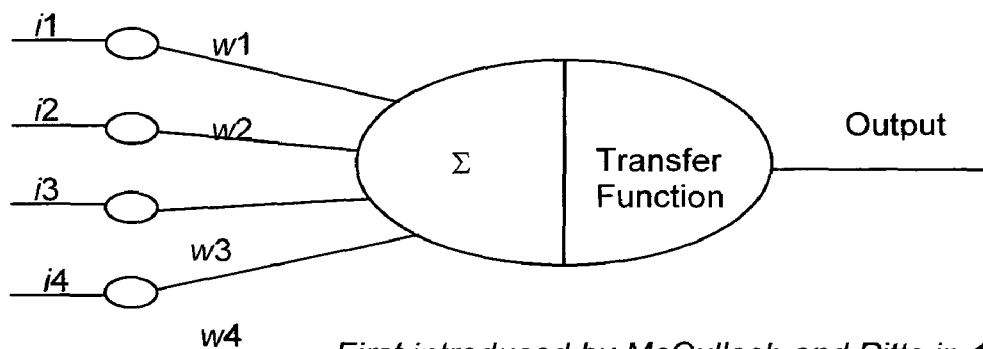
The brain itself, consists of approximately  $10^{14}$  neurons, each connected to  $10^4$  others [18]. Each nerve cell has defined inputs and outputs. The inputs to the cell are a set of fibres called dendrites, and the output is a long, branch like fibre called the axon. The connecting point between the axon of one cell and the

dendrite of another is called the synapse. The cells work by means of impulses which are generated by cells that 'fire', and are then passed along axons via synapses to dendrites of other cells which may in turn fire depending on the inputs [Web 5].

### 1.2.8.2 Artificial Neurons

This is the basic working model in about 90% of neural network applications. Each input  $i$  to the neuron has a weight  $w$  associated with it, the values are multiplied together to give the weighted input. (See Figure 1.7)

Figure 1.7 Diagram of an Artificial Neuron



*First introduced by McCulloch and Pitts in 1943*

The neuron calculates the sum of all the weighted inputs; this total is then modified by a transfer function, typically a smooth non-linear function of the total weighted input. Mostly, the functions used are the

Sigmoid function:  $\text{output} = 1 / (1 + e^{-\text{total weighted input}})$

Tanh function:  $\text{output} = a * \tanh (b * \text{total weighted input})$

where  $a$  = output range

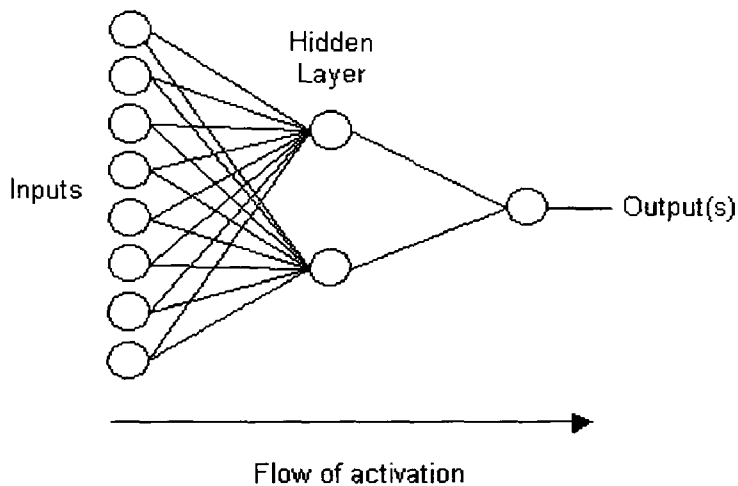
$b$  = slope of the curve

Finally, the output of the neuron is passed as input to other neurons [Web 5].

### 1.2.8.3 Artificial Neural Networks (ANNs)

The processing power of a neural network lies in the interconnections rather than the processors as with conventional computers. Networks consist of 3 or more 1-dimensional or 2-dimensional layers of neurons. (See Figure 1.8)

**Figure 1.8** Diagram of a Neural Network [Web 15]



In a layered network, the extent of the interconnections is considered. A fully connected network is where every neuron is connected to all other neurons, and a randomly connected network is where a random subset of interconnections are implemented. A partially connected network is where a subset of interconnections are made based on some rationale [Web 5].

#### *1.2.8.4 Training*

Neural networks are “trained” in order to solve problems, and the network appears to “learn” how to respond to a given set of inputs. There are two distinct methods of training a neural network, Supervised and Unsupervised, each requiring different types of networks.

Supervised training involves showing the network examples of what we want it to learn. It is told what the inputs are and what the expected output should be. About 90% of applications use this method. There are possibly hundreds or thousands of weights in a typical network and so manually setting these weights is not an option.

The neural net uses a learning algorithm to adjust the weighting factors so that the difference between the expected output and the actual output of the network (i.e., the error) is minimised. After the training of the network has been completed, the ANN can be used to model the system without further supervision [Web 5]. In order that a supervised network can solve a problem, the weights in the network are set so that the expected output is generated for a given set of inputs. Neural networks can be computationally intensive during training and may need thousands of iterations to reach a solution. In some cases, there is even no guarantee that an acceptable result will be found. This is a characteristic of ‘gradient descent’ problems.

Unsupervised training is where no information about what the data represents is given to the network. The network is just shown the data, which it attempts to make sense of. An example of unsupervised learning is Hebb

Learning, where, if two units are active at the same time, the strength of the connection between them is increased; otherwise the connection is made more inhibitory [Web 5].

#### *1.2.8.5 Testing*

Once the network has been trained, it needs to be tested using an unseen data set called the Test Set. This will indicate how well the network has been trained. If the network performance is poor, then retraining the neural network using a different data set maybe necessary [Web 5].

#### *1.2.8.6 The Expected Output*

It is useful to visualise what the expected output should be, by plotting it. If a single line can be drawn to separate the different values of the output, then the problem is said to be linearly separable, otherwise it is non-linearly separable. In most cases, the problems will be non-linearly separable and will need to separate out classes in multi-dimensional space. Here, the classes will not just be single point variables but groups of data [Web 5].

#### *1.2.8.7 Features of an ANN*

The network holds information / knowledge in its architecture and in its weights, where the information is distributed throughout the network. This opens up the possibility of fault tolerance. Consider any output neuron. Its activation is a



function of the sum of many inputs. If one of the inputs to that neuron is missing due to some fault then the network should still operate correctly.

Neural networks have the ability to cope with noisy input data. The output of the network is reliant on a complex sum of many inputs, therefore it is likely that no one input is particularly vital in generating the output. Thus, the effect of noisy inputs on the network performance is negligible [Web 5].

#### *1.2.8.8 Neural Networks and Data Analysis*

Neural networks outperform many methods of analysis because they can successfully :

- Be developed from data without an initial system model
- Handle noisy or irregular data
- Quickly provide answers to complex issues
- Be easily and quickly updated
- Interpret information from tens or even hundreds of variables or parameters
- Readily provide generalised solutions

#### *1.2.8.9 Applications of Neural Networks*

Their applications are almost limitless but they fall into several main categories.

## Classification

### **Business**

Credit rating and risk assessment, Insurance risk evaluation, Fraud detection

### **Engineering**

Machine vision, Speech recognition, Radar signal classification

### **Security**

Face recognition, Speaker verification, Fingerprint analysis

### **Medicine**

General diagnosis, Detection of heart defects

### **Science**

Recognising genes, Botanical classification, Phytoplankton identification

## Modelling

### **Business**

Prediction of share and commodity prices

### **Engineering**

Colour discrimination, Robot control and navigation

### **Science**

Prediction of the performance of drugs from the molecular structure

### **Medicine**

Medical imaging and image processing

## Forecasting

Future sales, Production Requirements, Market Performance

[Web 5]

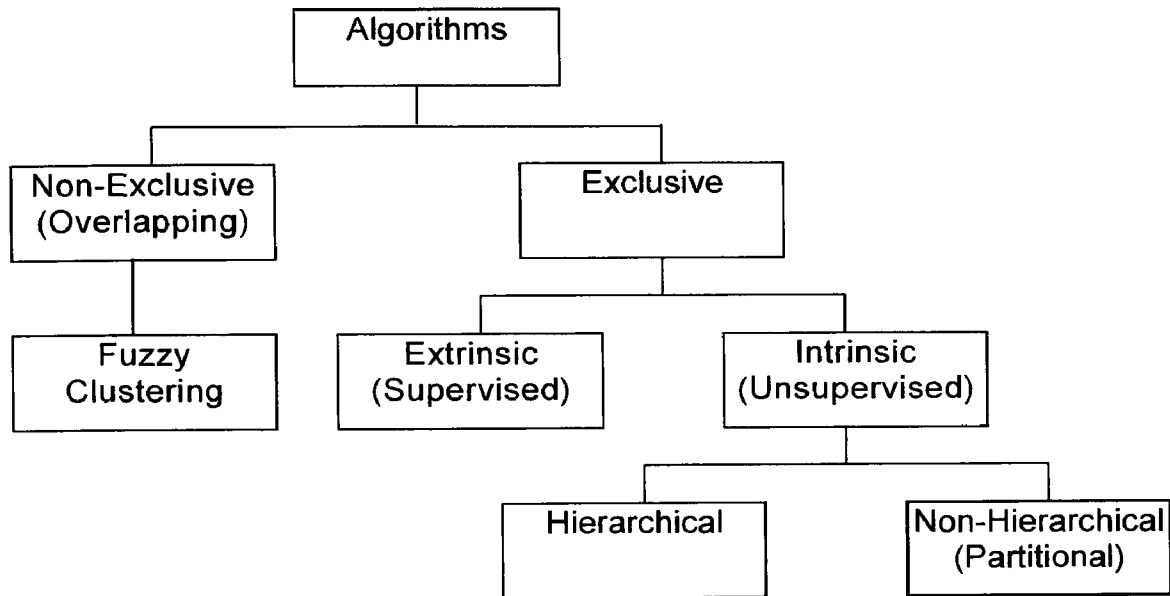
### **1.2.9 Clustering**

Clustering can be described as a technique, which categorises large data sets into smaller sets or clusters. Clustering algorithms use the raw information to obtain groups of data, which are similar in nature. Frequently data is represented as data points on a graph and in general, 'similarity' between two data points is determined by the Euclidean distance between two such points. Clustering Algorithms fall in two main categories, Exclusive and Non-Exclusive with further categories following from each, (see Figure 1.9) all are discussed below. There are many uses for clustering algorithms, classification of species, [Web 6]. This makes clustering an obvious technique to use for analysing flow cytometry data.

Clustering algorithms could be used to identify individual species of phytoplankton if the appropriate parameters are chosen for the flow cytometer to measure. However, in this research, clustering is used as a tool to classify the species. Here, instead of distinguishing individual species, clustering is used to group together the data representing the different species which have similar characteristics.

**Figure 1.9** Tree of Clustering Algorithm Types

This is based on the tree diagram by Lance and Williams 1967 [19].



#### *1.2.9.1 Exclusive and Non-Exclusive*

An exclusive classification is where objects are assigned to one and only one cluster, for example people grouped by their age. Non-exclusive classification in comparison, allows objects to belong to more than one cluster, for example, people grouped by the films category, as most people like a variety [20]. The following clustering techniques are of the 'exclusive' type only.

#### *1.2.9.2 Intrinsic and Extrinsic*

An intrinsic classification just uses the proximity matrix to cluster the data, whereas extrinsic classification also uses points in space, and relies on a 'teacher' to separate the objects. In pattern recognition, intrinsic classification is

also known as 'unsupervised learning', this is because no category labels are used. The category labels denote any "*a priori* partition of the objects". Suppose data on various health parameters were collected from smokers and non-smokers, then an extrinsic classification would look at discriminating non-smokers from smokers according to the variables, whereas an intrinsic classification would group the data according to similarities in the parameters to investigate if smoking was a likely cause of certain health problems. "Intrinsic classification is the essence of cluster analysis", so only this type of algorithm will be covered [20].

#### *1.2.9.3 Hierarchical and Non-Hierarchical*

Intrinsic classifications can be further separated into Hierarchical or Partitional based on the structure of the data used. Non-hierarchical is a single partition, compared to hierarchical which is a "nested sequence of partitions", and therefore, can be described as a "special sequence of partitional classifications."

Furthermore, there are several different implementations available for exclusive intrinsic hierarchical classification: -

- Agglomerative and Divisive
- Serial and Simultaneous
- Monothetic and Polythetic
- Graph Theory and Matrix Algebra [20].

#### 1.2.9.4 Hierarchical Clustering Algorithms

Hierarchical algorithms are methods, which group the data over several iterative steps until the required number of clusters is achieved, for example, the Minimal Spanning Tree algorithm.

This particular technique sets each data item to be a different cluster. Then iteratively, the two clusters which are closest together, i.e. most similar, are combined to create a new single cluster. This continues until the required number of clusters is obtained.

A list of objects arranged to show the clustering is not always easy to understand, so instead a diagram called a Dendrogram can be used. This is a tree-type structure, which illustrates the hierarchical clustering, with a layer of  $n$  nodes, one for each cluster, and lines to connect the nodes to show which are nested within each other [20]. (See Figure 1.10)

**Figure 1.10** Example of a Dendrogram

##### Clusterings

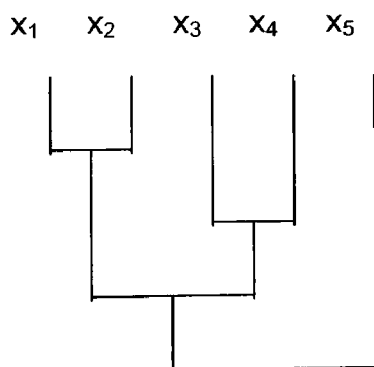
$\{ (x_1), (x_2), (x_3), (x_4), (x_5) \}$

$\{ ((x_1), (x_2)), (x_3), (x_4), (x_5) \}$

$\{ ((x_1), (x_2)), ((x_3), (x_4)), (x_5) \}$

$\{ (x_1, x_2, x_3, x_4), (x_5) \}$

$\{ (x_1, x_2, x_3, x_4, x_5) \}$



Hierarchical clustering is commonly applied to biological, social, and behavioural sciences, as taxonomies are prevalent in these areas [20].

#### *1.2.9.4.1 Agglomerative and Divisive*

Agglomerative and divisive methods operate in the same way for both hierarchical and partitional classifications. An agglomerative classification assigns each object as a cluster, then merges these clusters together until only one exists. (See Figure 1.10 for an example.) A divisive classification however, functions in the opposite way by starting with one large cluster and then divides this into smaller ones [20].

#### *1.2.9.4.2 Serial and Simultaneous*

Here, the serial method uses each pattern separately, compared to the simultaneous type which as its name suggests, uses the whole data set at the same time [20].

#### *1.2.9.4.3 Monothetic and Polythetic*

Classification algorithms can also be viewed as Monothetic and Polythetic. This relates to the number of parameters used simultaneously, so that monothetic uses the parameters one by one, and polythetic uses them all together. The following algorithms are only polythetic [20].

#### *1.2.9.4.4 Graph Theory and Matrix algebra*

Clustering algorithms can either be expressed in terms of graph theory, algebra, or in some cases both. Graph theory classifies data based on how complete the graph is or how fully connected it is, and an example of an algebraic method used for classifying is the mean-square-error [20].

#### *1.2.9.5 Non-Hierarchical Clustering Algorithms*

Non-hierarchical algorithms are methods where the number of clusters required is set at the start, a scoring system is used to allocate points to clusters and data are grouped together in order to optimise this clustering. These algorithms make use of centroids, central points which represent the cluster they are included in. One clustering criterion may be to minimise intra-cluster variance, which is calculated by summing the variance of each parameter that constitutes the centroid. An example of a Non-hierarchical algorithm is K-Means.

Here, the number of clusters required is set to be ' $n$ ' say. Then  $n$  data items are selected to be the initial clusters so that the data points have the greatest distance between them. Next, iteratively, each item in the population is considered and assigned to the cluster, which is closest to it. The centroid for each cluster is re-calculated each time a new data item is added to the cluster. This continues until the required number of clusters has been achieved, and all data items have been included [20].



#### *1.2.9.6 Fuzzy Clustering*

The general function of a scatter matrix based clustering algorithm is to group the data into  $k$  clusters, where each cluster is represented by a centre  $\mu_t$  and a scatter matrix  $S_t$ , with  $t = 1, \dots, k$ . As far as the membership  $u_{it}$  is concerned, if object  $i$  belongs completely to cluster  $t$ ,  $u_{it}$  is 1, if it does not belong to the cluster,  $u_{it}$  is 0. Further more, if  $u_{it}$  is said to be boolean and therefore only take the values 0,1, this is 'crisp' clustering. If however,  $u_{it}$  is able to take all values between 0 and 1, this is said to be 'fuzzy' clustering. In this research, five variants of a recently proposed multivariate clustering algorithm (fuzzy K-means with scatter matrices) were compared with each other and an earlier multivariate clustering algorithm used in flow cytometry, based on critical distances.

#### *1.2.9.7 Clustering Algorithms and Flow Cytometry Data*

The data sets involved in flow cytometry are large, in the order of  $10^4$  to  $10^6$ . This means that any clustering technique which involves calculating pairwise distances between patterns are not computationally feasible. This is because the number of calculations and required storage area is of the order  $N^2$  where  $N$  is the number of patterns, and the data set used in this research frequently numbered  $10^4$ , giving the number of calculations of the order  $10^8$ . Furthermore the clusters in each data set can vary quite considerably with regards to the size of the area covered by the cluster, and the actual number of points per cluster may also change over orders of magnitude.

Another problem that can be encountered is if the value of a parameter is outside the valid range. In such cases, the value is changed to a constant of between 0 and 1024 (for 10-bit data). Such data points are effectively limited to lying in one or more planes of the hypercubic region of data space. If clusters of data points like these lie in a plane, they have zero volume and are therefore degenerate, which means the clustering algorithm used must be able to cope gracefully with clusters that only occupy part of the full hypercubic area of the data space.

The density functions of each cluster are in general approximated closely by multivariate Gaussians as they are based on biological measurement data. Unfortunately, the clusters themselves can be elongated, with the length up to ten times the width. This means that any algorithm based solely on Euclidean distance metrics would fail to converge to a clustering solution.

#### *1.2.9.8 Distance Metrics*

There are several distance metrics that can be used to establish the similarity of a new object with regards to a cluster(s) of existing objects. The idea is to establish if the new object is near the mean of one of the clusters. If it is, the object can be classified as being in the same cluster, and conversely if it is far away, the object does not belong to that cluster. To determine if an object is near or not, merely looking at the data is not an option. This is because real life data sets are multi-dimensional and therefore cannot be visualised. Consequently, a

mathematical equation is used instead. Two such equations are considered here, Euclidean and Mahalanobis.

#### *1.2.9.8.1 The Euclidean Distance Metric*

Using this distance metric, data objects are clustered by finding the nearest cluster to the new object.

Firstly, this method has no measurement of how successful or otherwise, a new object is in matching any existing clusters. Secondly, this metric merely measures a “relative distance from the mean point in the group” [Web 9], the distribution of the data within the cluster is ignored. Finally, with the Euclidean distance metric, the variation of the data in all the dimensions is not taken into account.

In clustering applications a basic minimum-distance classifier like the Euclidean may be suitable. However, such classifiers frequently have unacceptably high error rates. This may be due to one of several reasons:

- High correlation of the data
- A non-curved decision boundary
- Too much complexity in the feature space
- The presence of sub-classes

##### *1.2.9.8.1.1 Correlated Features*

Two or more features are said to be correlated if they are affected by a common factor and therefore change simultaneously. For example, the area and volume of a shape will both vary in relation with the length and width.

Correlation has a detrimental effect on the performance of the clustering algorithm, so that an object at one extreme of a cluster can be nearer to another cluster than its own. This can also be seen if the units of measurement used for the different variables are not scaled very well, for example, if one variable is measured in microns and another in kilometres. One way to solve this problem is to use the Mahalanobis metric. (The Mahalanobis Metric is covered in section 1.2.9.8.2)

#### *1.2.9.8.1.2 Curved Boundaries*

A Euclidean distance classifier will only have linear boundaries, which may not be able to cope with rate of growth of some variables. This can occur if one variable increases linearly, while another increases quadratically. The result of this would be a distorted feature space, which would reduce the success of the clustering. Possible solutions to this are:

- Redesign the data set so all features grow linearly.
- Use the Mahalanobis distance metric which can produce quadratic boundaries
- Use a neural network ( this approach has been considered in previous research)

#### *1.2.9.8.1.3 Complex Feature Space*

It is not always the variability in the data and therefore 'noise' that causes clustering problems, occasionally it is complexity of the data. This can happen if the data set being clustered is significantly more complex than the classifier.

#### 1.2.9.8.2 *The Mahalanobis Distance Metric*

With the Mahalanobis metric, the standard deviation of the object from the cluster mean is used as the unit of measurement. This means that the ellipse-shape perimeter of the cluster is the one standard deviation boundary. The result of this is that a statistical probability can be placed on the success or otherwise of the new object matching the existing ones.

This measurement is concerned with the variation between data of one species (variance), as well as the variation between the species (co-variance). Variation in the data is allowed for by weighting the differences towards the cluster mean, and not just treating the values equally.

“Just like many multivariate quantitative methods, the Mahalanobis distance can solve for multiple dimensions simultaneously” [Web 9].

Unfortunately, there can be some drawbacks to this method. As the number of variables increases, the amount of time and memory needed doesn't increase linearly, but quadratically. Also there can be difficulties in calculating accurate covariance matrices. But these issues are really only significant if there is a large number of variables involved [Web 10], in the Mahalanobis case, the limit is between ten and fifteen, so the data used here which has seven variables, falls well within the limits.

Having a large number of items in a data set can cause problems with overfitting by the Mahalanobis metric. Matching the objects is just as important as the variations between objects, and this can result in matching objects not being classified in the correct cluster.

## Chapter 2 - Literature Survey

### 2.1 Why use Clustering Algorithms?

The various light scatter, diffraction, and fluorescence parameters measured by AFC can provide characteristic 'signatures' for each microbial cell, which allow taxa to be discriminated with the use of pattern-recognition techniques such as ANNs [21] [13] [19] [22] [23] [24]. More than 70 species have been identified successfully by ANNs trained on AFC data obtained under pure cultures of marine microalgae grown under controlled conditions in the laboratory [19]. However, species growing in the field are likely to show greater variability in size, shape and pigmentation due to a multitude of environmental factors [25], thus producing a corresponding increase in the variability of the AFC signatures. ANNs require training on a representative sample of each species that is to be recognised, and unless the training data reflect such biological variation, ANN analysis of field samples will not be reliable.

The large data sets generated by AFC pose a number of challenges for analysis. The data sets are often multidimensional, making visual analysis by two-dimensional or three-dimensional scatter plots difficult, even with the use of dimensionality reduction techniques such as principal components analysis [Web 8]. Recourse is generally made to some form of clustering algorithm capable of exploiting the full multivariate nature of the data. However, the typical size of AFC data sets ( $>10^4$  patterns) precludes application of many standard clustering algorithms such as pair-group methods that rely on calculation of distances

between pairs of points. AFC clusters are frequently highly elongated, and the variance-covariance structure can differ considerably between clusters. Some clusters are large and sparse, whereas others are compact and dense and perhaps even degenerate (possessing zero variance in one or more dimensions and therefore zero volume) [2].

## **2.2 Fuzzy Clustering Algorithms and Applications**

The concept of fuzzy sets was first proposed by Zadeh [26] in 1965 and introduced the idea that a problem space could be blurred, which gave a different outlook on how to solve pattern recognition problems. The use of fuzzy sets in clustering was proposed by Bellman et al [27] and led to the first fuzzy clustering algorithm, developed in 1969 by Ruspini [28] which introduced fuzzy criterion functions.

Clustering algorithms such as the K-means are based on the distances between data point and cluster centre and it is well documented [29] [20] [30] that this criterion is only effective when the clusters are spherically-based with similar volumes. Gustafson and Kessel developed a method based on the Fuzzy K-Means algorithm [31], which tried to accommodate ellipsoidal clusters by use of the Mahalanobis distance metric instead of the Euclidean metric, and scatter matrices. This performs better with ellipsoidal clusters compared with spherical clusters, but the algorithm again is only effective when the clusters have similar volumes. Even though it will function when the data is grouped into clusters of different shapes, their relative sizes must be known.

The product of fuzzy determinants criterion developed by Trauwert et al avoids this restriction, but is more likely to find alternative clustering solutions to the natural clusters [32]. Trauwert proposed an algorithm based on the total volume of the clusters, which was measured by the sum of the square root of the determinants of the fuzzy covariance matrices [33]. The classical algorithms: fuzzy K means, fuzzy product of determinants, and minimum fuzzy volume were investigated and their performances compared with that of the minimum total hypervolume algorithm. When the clusters are not too dissimilar with regards to shape, size and cardinality, the 'new' algorithm performed comparably with the existing algorithms. However with a large variation in the clusters based on the above characteristics, "the minimum total hypervolume method appears to be the only algorithm able to reproduce the expected natural clusters" [33]. The product of fuzzy determinants criterion is the same as the Maximum Likelihood algorithm used in this study.

Instead of basing an algorithm on scatter matrices, Krishnapuram and Kim explored the use of a determinant or volume criteria for clustering [34]. They derived the minimum scatter volume (MSV) algorithm that minimises the scatter volume of the clusters, and the minimum cluster volume (MCV) that minimises the sum of the volumes of the individual clusters. The determinant of the sum of cluster scatter matrices was proposed as a possible clustering criterion by Duda and Hart [29]. Krishnapuram and Kim derived an iterative algorithm that minimises this criterion, by fuzzifying it. They devised both crisp and fuzzy versions of the sum-of-volumes criterion, and derived the minimum cluster



volume (MCV) algorithm from this. The performance of the minimising volume criteria algorithm was compared with that of traditional algorithms. The results illustrated that “in general MCV gives better results than the K-means, MSV, and Gustafson-Kessel algorithms. MCV is also quite versatile and can be used in a variety of applications such as segmentation of intensity / range images and classification” [34].

### ***2.2.1 From a fuzzy clustering in flow cytometry point of view***

Demers et al proposed a development of the k-means algorithm, the idea of which was to try and solve the elongated cluster problem. A scatter matrix for each cluster, estimated from the data points around the cluster centre, was used as a more general distance measure to include details of the cluster shapes. The algorithm as with all iterative methods, can result in poor clustering solutions if it becomes trapped in a local minimum, rather than the global minimum. When this happens, the user is required to identify and reject these deficient results [35].

The algorithm was made more robust to being trapped in local minima, as instead of the data points only belonging to the closest cluster, they are allowed to belong to all clusters to some degree. Rousseeuw et al [36], introduced several extensions of this fuzzy k-means approach using scatter matrices to include details of the cluster shapes, with various different cost functions. These are some of the algorithms that have been investigated in this research [2].

One drawback of these algorithms is that it is necessary to specify  $k$  the number of clusters in the data *a priori*. So, it would be more desirable if the algorithm used could determine the ‘natural’ number of clusters from the data.

This can be achieved not only “by overspecifying the number of clusters and removing those that according to some criterion become redundant during the course of the algorithm [37]” [38]. But also where a “stability criterion” of the clusters is defined as a function of a scale parameter and is used to decide the optimal data partition [39]. This was considered as future research.

### ***2.2.2 Applications of Fuzzy Clustering***

Starting with Sneath’s work on bacteriological taxonomy, partitioning algorithms have been applied to many applied science areas [40]. The large range of applications of these algorithms include, “psychology, sociology, geology, medicine, experimental particle physics, operations research, and the technology of automatic reading machines” [41]. Fuzzy clustering has been applied in numerous areas, for example, pattern recognition with satisfactory results and “considerable economic benefit” [42].

An important benefit of using fuzzy clustering is that it is an unsupervised method which has no need for training data. One of the most significant issues in pattern recognition is feature extraction. Therefore, a more important function of fuzzy clustering is that it can not only “extract features from raw data directly [43], but also select the optimal feature sets or reduce the dimensionality of obtained features [44]” [42].

## **2.3 Summary of Literature Survey**

In summary, it is well documented [29] [20] [30] that the K Means method is only effective when the clusters are spherically based with similar volumes,

and as the data used in this study has clusters of various shapes, it is not suitable. Also, clustering algorithms such as the K-means are based on the distances between data point and cluster centre measured using the Euclidean metric. This is adequate for data of low dimensionality, but as our data has seven dimensions, it is not computationally feasible to use this approach. The Gustafson and Kessel algorithm [31], although it can successfully partition the data into ellipsoidal clusters by use of the Mahalanobis distance metric and scatter matrices. It is again, only effective when the clusters have similar volume, which is not the case with the data set used in this research. Trauwert avoids this issue, using a product of fuzzy determinants criterion, but this tends to provide a clustering solution not based on the natural clusters [32]. Krishnapuram and Kim explored the use of a determinant or volume criteria for clustering [34]. They derived the minimum scatter volume (MSV) algorithm that minimises the scatter volume of the clusters, and the minimum cluster volume (MCV) that minimises the sum of the volumes of the individual clusters. "In general MCV gives better results than the K-means, MSV, and Gustafson-Kessel algorithms. MCV is also quite versatile and can be used in a variety of applications such as segmentation of intensity / range images and classification" [34]. However, as Rousseauw proposed five variations of a scatter matrix fuzzy clustering method [36], it was decided to investigate those approaches as it would be more efficient to implement these as they are all based on the same general algorithm, and include the Maximum Likelihood algorithm which is the same as the product of

fuzzy determinants criterion. Further details of the Rousseeuw methods are given later.

Gustafson and Kessel developed a method based on the Fuzzy K-Means algorithm [31], which accommodated ellipsoidal clusters by use of the Mahalanobis distance metric and scatter matrices. It will function when the data is grouped into clusters of different shapes, even though their relative sizes must be known. This is the same as Rousseeuw's Adaptive Distances algorithm used in this research, see further details of this method later.

Trauwert proposed an algorithm based on the total volume of the clusters [33], which was measured by the sum of the square root of the determinants of the fuzzy covariance matrices. When the clusters are not too dissimilar with regards to shape, size and cardinality, the 'new' algorithm performed comparably with the existing algorithms. However with a large variation in the clusters based on the above characteristics, "the minimum total hypervolume method appears to be the only algorithm able to reproduce the expected natural clusters" [33]. As the data used in this study produce clustering solutions where clusters have various shapes, sizes, it was decided, this was an appropriate algorithm to study. The product of fuzzy determinants criterion is the same as the Maximum Likelihood algorithm developed by Rousseeuw, used in this research.

Rousseeuw et al [36], introduced several extensions of a fuzzy k-means approach using scatter matrices to include details of the cluster shapes, with various different cost functions. The idea of this was to solve the elongated cluster problem. It is, robust to being trapped in local minima, as instead of the

data points only belonging to the closest cluster, they are allowed to belong to all clusters to some degree. The Mahalanobis metric is used, therefore calculating distances is computationally feasible. These are scalable algorithms, as scatter matrices always implemented as two-dimensional array, not matter how many data points are used, or the dimensionality of the data. The data set used in this research contains  $10^4$  data points of seven dimensions, therefore it was decided that the different variations developed by Rousseauw [36] were suitable to be investigated.

## 2.4 Real Water Samples

'Real water' samples are used in flow cytometry research and initially would probably contain other organisms like bits of dead fish, food remnants, bacteria etc., as well as phytoplankton. As the water sample flows through the flow cytometer, the volume of the water flowing is reduced until it becomes a microscopic drop of water containing a single organism. The phytoplankton in this study are only between  $1\mu$  and  $42\mu$  long, so any 'foreign bodies' in the water sample could only potentially cause problems if they were also that small.

In addition to this, the variables measured using the flow cytometer, were specifically chosen by researchers at Plymouth Marine Labs, to be able to identify and/or classify plant plankton in water samples. The database itself holds on average  $10^3$  records on each of 63 species and therefore it is very likely that the data patterns will represent some or all of the stages in a plankton lifecycle. Consequently, a new data pattern could be compared with all existing records

prior to cluster analysis, as any measurements recorded about other particles will most likely not match any of the data already collected. If it is not similar to any records, then more analysis could be done to make sure it isn't data representing a 'new' species of phytoplankton to the data set.

It is also possible, that the water sample could contain no particles. As some of the variables recorded involve size, if there was no particle in the water stream to measure, the data recorded would be negligible or even zero for area and width as there would be nothing to measure. Data patterns like these could be identified and removed from a data set prior to cluster analysis.

## **2.5 How many groups are there really in our data?**

As there are many microscopic particles in water samples, could it be possible that there are more groups than just the ten or so allocated for phytoplankton?

Some data is multi-modal, so it maybe that there are more than 10 groups of data in the data sets. This is because data for one species can be partitioned into more than one cluster, especially if it is a chain forming species of phytoplankton. The phytoplankton are so small that if two, three or more attach themselves together they go through the flow cytometer as a single particle and therefore chains of different lengths will have different measurements and be classified into different clusters. Therefore it is likely that there are more than ten groups of data in our data. However, as scientists look for certain characteristics

of phytoplankton, it is not important how many groups of data are in the sample, just if there are any that exhibit these features.

## **2.6 Do we need to redefine the taxonomic groups?**

Flow cytometry data of phytoplankton representing one species can be clustered as though it is data representing a different species. This is because the measurements from the different species, could have been taken at different times in their lifecycles so that the measurements appeared to represent the same species, and therefore they could be partitioned into the same cluster. As a result of this, it could be argued that the taxonomic groups should be changed in order to cater for this anomaly.

In this case, I don't think redefining the taxonomic groups would help at all because at other times in the phytoplankton lifecycle, the data for the two species could be classified separately. If they were redefined, it is likely that the same problem would occur but for data from other species. Also because, one method of getting around this, could be to use all the data representing all the species in a data set generated. The idea of this is that, most of the data recorded should partition into one cluster, with only a relatively small proportion partitioned into different clusters.

Most importantly, the measurements that are recorded in the AFC data, aren't chosen so that the phytoplankton can be taxonomically identified. As mentioned before, they are chosen so that scientists can analyse the AFC data looking for specific characteristics, for example, do the phytoplankton have red

fluorescence, and where knowledge of the phytoplankton species is not important.



## Chapter 3 – Materials and Methods

### 3.1 Data sets

A data set of approximately  $1.1 \times 10^6$  records of 7-parameter FACScan flow cytometer data (see Table 1.3 for the parameters) covering 64 marine phytoplankton species grown in culture was used here [45]. (See Table 3.1)

**Table 3.1** List of all phytoplankton species in full data set

<i>Alexandrium tamarense</i>	<i>Amphora coffaeiformis</i>	<i>Amphidinium carterae</i>	<i>Aureodinium pygmentosum</i>
<i>Chaetoceros calcitrans</i>	<i>Chlorella salina</i>	<i>Chroomonas sp.</i>	<i>Chroomonas salina</i>
<i>Chrysochromulina camella</i>	<i>Chrysochromulina chiton</i>	<i>Chrysochromulina cymbium</i>	<i>Chrysochromulina polylepis</i>
<i>Chlamydomonas reginae</i>	<i>Cryptomonas appendiculata</i>	<i>Cryptomonas calceiformis</i>	<i>Cryptomonas maculata</i>
<i>Cryptomonas reticulata</i>	<i>Cryptomonas rostellata</i>	<i>Dunaliella minuta</i>	<i>Dunaliella primolecta</i>
<i>Dunaliella tertiolecta</i>	<i>Emiliana huxleyi</i>	<i>Gymnodinium micrum</i>	<i>Gymnodinium simplex</i>
<i>Gymnodinium veneficum</i>	<i>Gymnodinium vitiligo</i>	<i>Gyrodinium aureolum</i>	<i>Hemiselmis brunnescens</i>
<i>Hemiselmis</i>	<i>Hemiselmis</i>	<i>Heterocapsa</i>	<i>Isochrysis</i>

<i>rufescens</i>	<i>virescens</i>	<i>triquetra</i>	<i>galbana</i>
<i>Micromonas</i> <i>pusilla</i>	<i>Nephroselmis</i> <i>pyriformis</i>	<i>Nephroselmis</i> <i>rotunda</i>	<i>Ochrosphaera</i> <i>neopolitana</i>
<i>Ochromonas</i> <i>sp.</i>	<i>Pavlova</i> <i>lutheri</i>	<i>Prorocentrum</i> <i>balticum</i>	<i>Phaeocystis</i> <i>pouchetii</i>
<i>Pseudopedinella</i> <i>sp.</i>	<i>Pelagococcus</i> <i>subviridis</i>	<i>Pheodactylum</i> <i>tricomutum</i>	<i>Plagioselmis</i> <i>punctata</i>
<i>Pleurochrysis</i> <i>carterae</i>	<i>Prorocentrum</i> <i>micans</i>	<i>Prorocentrum</i> <i>minimum</i>	<i>Prorocentrum</i> <i>nanum</i>
<i>Porphyridium</i> <i>pupureum</i>	<i>Prymnesium</i> <i>parvum</i>	<i>Pyramimonas</i> <i>grossii</i>	<i>Pyramimonas</i> <i>obovata</i>
<i>Rhodella</i> <i>maculata</i>	<i>Rhodomonas</i> <i>sp.</i>	<i>Scrippsiella</i> <i>trochoidea</i>	<i>Skeletonema</i> <i>costatum</i>
<i>Stichococcus</i> <i>bacillaris</i>	<i>Tetraselmis</i> <i>impellucida</i>	<i>Tetraselmis</i> <i>striata</i>	<i>Tetraselmis</i> <i>suecica</i>
<i>Tetraselmis</i> <i>tetrathele</i>	<i>Tetraselmis</i> <i>verrucosa</i>	<i>Thalassiosira</i> <i>weissflogii</i>	

From this data set two artificial data sets A and B were constructed, each consisting of  $10^3$  data patterns selected randomly for each of ten species. Both data sets thus contained a total of  $10^4$  data patterns. Data set A comprised ten species for which the corresponding AFC clusters did not overlap appreciably, as determined by examination with PCA and visualisation by 3D scatterplots, while

data set *B* comprised ten species for which the corresponding AFC clusters overlapped considerably. (See Table 3.2)

To make the data set more realistic in terms of real world data, the results were recorded during different times in the phytoplankton lifecycles so that normally unrelated species can be seen to have similar properties and may therefore be clustered together.

**Table 3.2** Data sets showing the species of phytoplankton used

<b>Non-Overlapping</b>	<b>Overlapping</b>
<i>Chrysochromulina chiton</i>	<i>Amphora coffaeformis</i>
<i>Cryptomonas appendiculata</i>	<i>Amphidinium carterae</i>
<i>Emiliana huxleyi</i> b111	<i>Aureodinium pigmentosum</i>
<i>Hemiselmis brunnescens</i>	<i>Chrysochromulina chiton</i>
<i>Micromonas pusilla</i>	<i>Cryptomonas appendiculata</i>
<i>Nephroselmis pyriformis</i>	<i>Emiliana Huxleyi</i> b111
<i>Porphyridium pupureum</i>	<i>Gymnodinium veneficum</i>
<i>Rhodella maculata</i>	<i>Hemiselmis brunnescens</i>
<i>Rhodomonas</i> sp.	<i>Heterocapsa triquetra</i>
<i>Tetraselmis tetrathele</i>	<i>Nephroselmis pyriformis</i>

### 3.2 Experimental Procedure

In order to test the clustering algorithms, it was necessary to define the number of plankton species in the data *a priori*. Both data sets comprised of ten

species and therefore it was reasonable to expect ten clusters, however, with a random water sample, the number of species in the sample would not be known, so it was decided to use six and fourteen clusters as well. Using the AimsNet software [46], each algorithm was applied five times on each data set for six, ten, and fourteen clusters. All results were assessed for reliability and performance, as described below, as well as using the summarised results themselves.

### **3.3 Computer Hardware and Software**

The experiments were run on a Pentium 3 600MHz Viglen computer using two pieces of bespoke software:

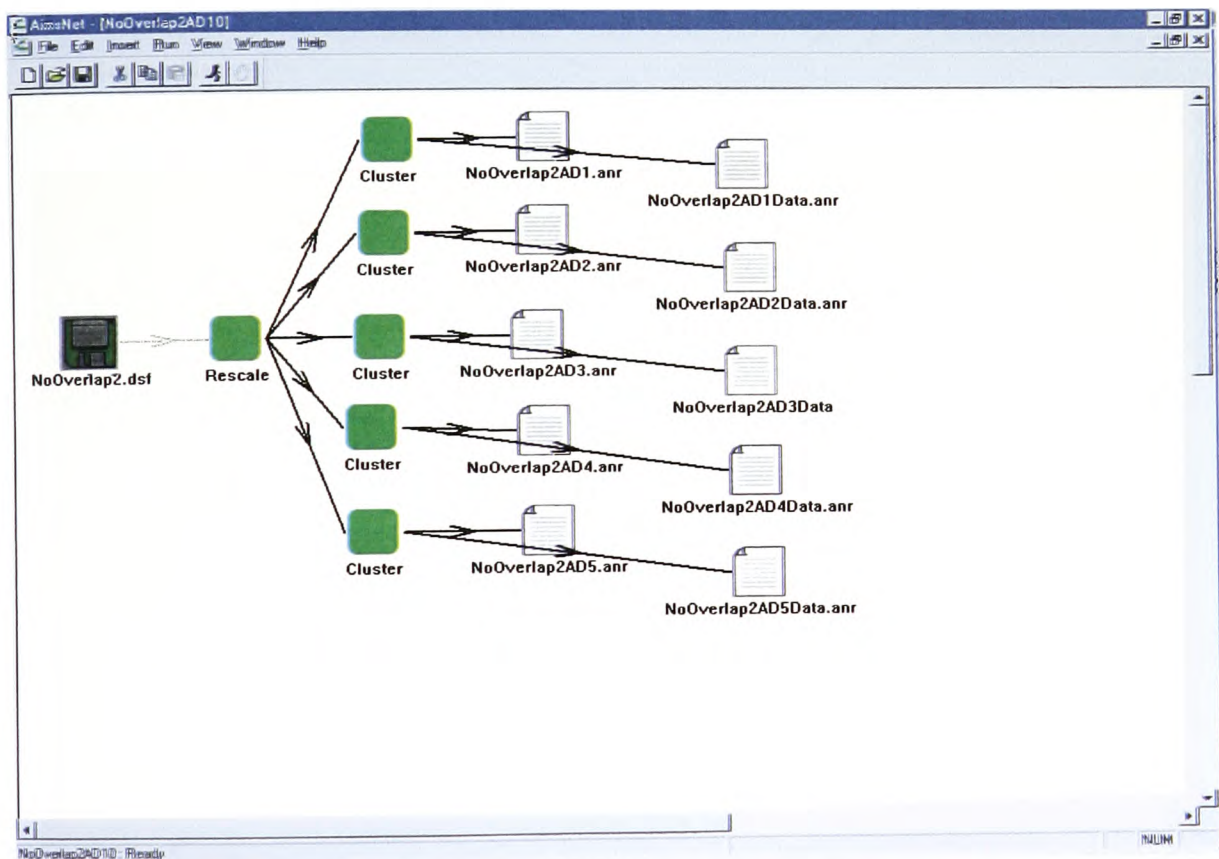
- i) AimsNet – developed under the Aims project, Automated Identification and Characterisation of Microbial Populations (CEC grant no. MAS3 – CT97 – 0080) - written by Malcolm F Wilkins.

“AimsNet” is a Windows™ application that brings the power of Artificial Neural Networks (ANNs) to the analysis of AFC data.

With AimsNet, the user constructs a connected structure of simple data processing modules to describe the desired data processing steps. Data flows from one module to the next via directional connections. The data, in either ASCII or FCS 2.0 formats, are read from file by a data source module. (FCS 2.0 is the standard flow cytometry data file format, adopted by the Society of Analytical Cytology. It allows data recorded on one instrument to be read for analysis on another computer [47].)

Processed data are written back to disk as a text file, by a data sink module. In between, data processing modules can be used to perform data rescaling, removal of parameters, re-ordination methods (principal components analysis, canonical variate analysis), cluster analysis, and pattern recognition via radial basis function networks” [46]. (See Figure 3.1)

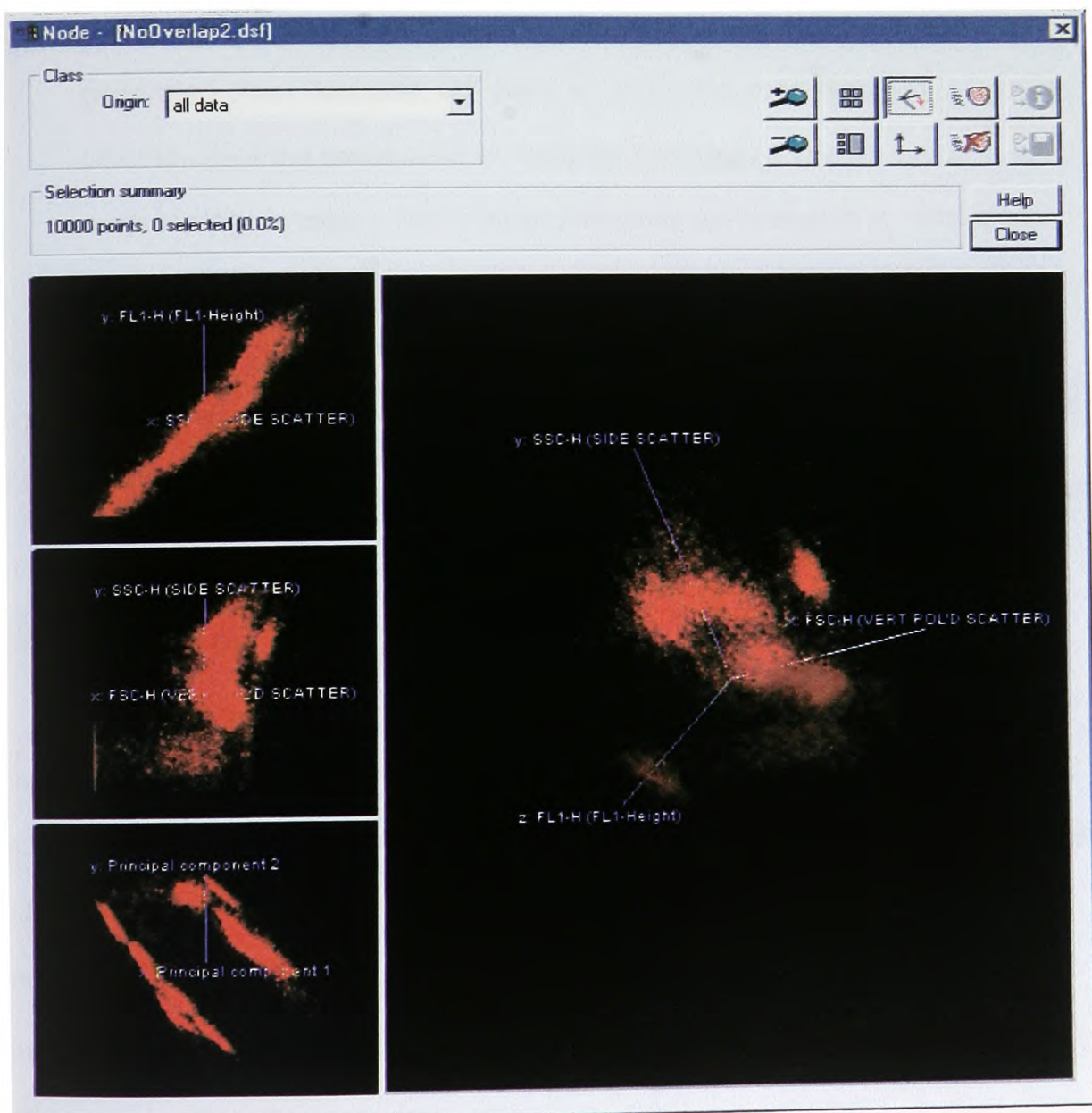
**Figure 3.1** AimsNet Front End



“Although powerful, AimsNet is designed throughout to be easy to use, even by those with limited knowledge of ANN’s.

AimsNet contains an integrated data viewer allowing visualisation and interpretation of the results of analysis via multiple interactive two dimensional and three dimensional scatterplots” [46]. (See Figure 3.2 below)

**Figure 3.2** AimsNet Data Viewer



AimsNet uses a large library of mathematical functions and is significantly object oriented, which exploits the functions of the C++ programming language.

- ii) Cluster Comparison software - written by Malcolm F Wilkins for use in this project. The software is run from the Windows™ command line, and compares the results of two clusterings. Here, two sink text files of data generated from AimsNet are read in and compared data point by data point to see if the same point in each file is assigned into the same cluster, a percentage showing the level of matching assignments is displayed on the screen [2].

Both pieces of software were written in C++ using Microsoft Visual Studio version 6.0.

### 3.4 Comparing the Results of Two Clusterings

An objective method is required for comparing the performance of two clustering algorithms. The result of running a clustering algorithm on a set of data is a partitioning of the data space; when this partition is applied to the data, a “clustering scheme” is generated for the data. Given two clustering schemes  $C$  and  $D$ , the proposed measure of similarity between them  $m(C, D)$  is defined as the *a priori* probability that two randomly-selected points drawn from the data set will be clustered in the same way under both clustering schemes; i.e. that either

both clustering schemes will assign both points to the same cluster, or that both clustering schemes will assign them to separate clusters. Letting  $c$  be the event 'both points are assigned to the same cluster under  $C$ ', letting  $d$  be the event 'both points are assigned to the same cluster under  $D$ ', letting  $c'$  be the event that 'both points are assigned to different clusters under  $C$ ' and letting  $d'$  be the event that both points are assigned to different clusters under  $D$ ', we can write  $m(C, D) = p(c)p(d|c) + p(c')p(d'|c')$ . The characteristics of this similarity measure are:

- (i) if  $C$  and  $D$  are identical (perfect correlation),  $p(d|c) = p(d'|c') = 1.0$  so  $m(C, D) = 1.0$ ;
- (ii) If  $C$  and  $D$  are independent (no correlation)  $p(d|c) = p(d)$  and  $p(d'|c') = p(d')$  and so  $m(C, D) = p(c)p(d) + p(c')p(d')$ ;
- (iii) It is commutative; i.e.  $m(C, D) = m(D, C)$ .

Because of the large size of the data set, determining  $m(C, D)$  by exhaustive enumeration of all possible pairs of data points is computationally intensive; however the proportion can be estimated to a desired level of accuracy, and confidence bounds placed on the value obtained, by repeatedly randomly drawing a large number  $n$  of pairs of data points and finding the proportion  $p$  of such trials for which the two schemes agree. The sampling standard deviation of  $p$ , a measure of the size of the uncertainty in the value of  $p$  as an estimator of  $m(C, D)$ , is given by  $q = \sqrt{np(1-p)}/n = \sqrt{p(1-p)}/n$  [2]. All



similarity measures reported in this paper were evaluated from  $n=10^7$  pairs of data points.

### **3.5 Evaluating Algorithm Performance**

Each algorithm was applied five times on both the non-overlapping data set and the overlapping data set, for  $k$  clusters where  $k = 6, 10$  and  $14$  clusters. Each algorithm was evaluated according to :

- (i) Consistency of results between multiple replicate runs of the algorithm for the same value of  $k$ , which was assessed by averaging the similarity measured between each pair combination of clustering schemes from the five replicate results;
  
- (ii) Performance of the algorithm in comparison to a “gold standard” clustering scheme which corresponds to the actual identity of the data patterns, which is known in advance in this project. This is applicable for  $k = 10$  clusters only. This was assessed by averaging the similarity measured between the clustering schemes from each of the five replicate results obtained and the “gold standard” clustering scheme.

## Chapter 4 – Clustering Algorithms

### 4.1 Introduction

The algorithms investigated here are based on the K Means Algorithm. This algorithm has been extended by various workers to include the use of scatter matrices (variance-covariance matrices), thereby allowing modelling of data with non-spherical clusters. Rousseeuw *et al.* unified this work by proposing four variants of a generic fuzzy  $k$ -means (FKM) algorithm: Adaptive Distances (AD); Maximum Likelihood (ML); Minimum Total Volume (MTV) and Sum of All Normalised Determinants (SAND). Each variant is designed to minimise a different objective criterion [36] (a measure of clustering ‘goodness’).

These algorithms require the number of clusters in an unknown data set to be determined *a priori*. Although, some algorithms exist which calculate the optimum number of clusters in a data set, they were not implemented and analysed in this research as the algorithms selected were fast, robust classifiers. It is at least as efficient to apply a fast robust algorithm several times in order to approximate the number of clusters in an unknown data set, than to use an untested algorithm to try to determine the optimum number.

“The four algorithms proposed by Rousseeuw *et al.* [36] are all variant forms of one generic fuzzy clustering algorithm, summarised below. In this algorithm, a cluster  $t$  is characterised completely by the position of its centre  $\mu_t$  and a scatter matrix  $\hat{S}_t$ . The distance  $d_{it}$  between the  $i$ th data pattern  $x_i$  and the

cluster centre is defined by the Mahalanobis generalised distance metric

$$d_{ii}^2 = (\mathbf{x}_i - \boldsymbol{\mu}_t)^T (\hat{\mathbf{S}}_t)^{-1} (\mathbf{x}_i - \boldsymbol{\mu}_t).$$

Each data pattern is associated to some extent with all the clusters, not just the closest; the extent of this association is given by  $u_{it}$ , the fuzzy membership of the  $i$ 'th data pattern to cluster  $t$ . The cluster memberships are subject to the constraints that  $0 \leq u_{it} \leq 1$ ,  $\sum_t u_{it} = 1$ ; in other words for any data pattern, the sum of its cluster memberships over all clusters is one. Constraining  $u_{it}$  strictly to the values 0 and 1 produces a crisp clustering, for which each data pattern is associated exclusively with only one cluster; however allowing  $u_{it}$  to take intermediate values between 0 and 1 results in a fuzzy clustering" [2].

The 3D scatterplot of the clusters (See Figure 1.6) indicated that the overlapping data set had five clusters, four separate clusters and one large cluster of six overlapped clusters. In view of this, extra experiments were run just on the overlapping data set whereby the data was adjusted so that the large conglomerate cluster represented one taxon of species, with the remaining species each designated as a taxon. Each algorithm was applied a further five times for five clusters with the adjusted overlapping data set.

Through experimentation, and a compromise between speed of convergence and accuracy of clustering, it was decided that the stopping criteria  $\varepsilon$  indicating convergence, would be set at 0.01.

## 4.2 Fuzzy K-Means FKM

Fuzzy  $k$ -means clustering is a variant of classic  $k$ -means clustering that allows data points to become associated to some degree with all clusters and not just the closest cluster [48] [49]. This reflects the inherent uncertainty in allocating a data point to a single cluster where there are several potential candidates. The algorithm is in practice stable and robust, and less likely than the classic  $k$ -means algorithm to produce inadequate clustering solutions. The purpose of the Fuzzy K-Means technique is to minimise the sum of the distances from the objects to the cluster centres. When the algorithm was implemented in AimsNet [46], it was extended to include the use of covariance matrices, allowing non-Euclidean distance metrics.

The variant used in this study is obtained by changing the manner of calculating the quantities  $A_i$  and  $B_{it}$  which are variables used to represent the relevant objective function. The alternative Fuzzy K-Means algorithm follows the notation in the paper by P. J. Rousseeuw published in 1996 so that [36]:

$$A_i = 0, \text{ and}$$

$$B_{it} = e_1 e_2 (\mathbf{x}_i - \boldsymbol{\mu}_i)' (\hat{\mathbf{S}}_i)^{-1} (\mathbf{x}_i - \boldsymbol{\mu}_i)$$

where  $e_1$  and  $e_2$  are the absolute values of the largest two eigenvalues of the fuzzy covariance matrix  $S_i$ . This is because, with degenerate clusters, at least one of the small eigenvalues would be close to zero. If this was used,  $B_{it}$  would become zero as the matrix determinant is a product of all the eigenvalues. Therefore by using the two largest eigenvalues, the algorithm remains stable even when degenerative clusters are encountered. The focus of this method is to

minimise the sum of the squared distances from the objects to the cluster centres. This is achieved by using equations  $A_t$  and  $B_{it}$  above which adapts the general algorithm to the FKM approach.

The variant of the Fuzzy K Means algorithm used is as follows:

1. For each cluster  $t$  initialise the cluster centre  $\mu_t$  to a randomly-selected data pattern and initialise the scatter matrix  $\hat{S}_t$  to the identity matrix
2. For each cluster  $t$ , calculate  $A_t$  and  $B_{it}$  which are variables used to simplify the relevant objective function where,

$$A_t = 0, \text{ and}$$

$$B_{it} = e_1 e_2 (x_i - \mu_t)' (\hat{S}_t)^{-1} (x_i - \mu_t)$$

3. Calculate memberships: for each data pattern  $x_i$ :
  - (i) Initialise  $T_i$  to an empty set where  $T_i$  is a set of indices  $t$  of the clusters that data pattern  $x_i$  belongs to,
  - (ii) For each cluster  $t$  calculate the distance from each data pattern  $x_i$  to the cluster centre  $B_{it} \leftarrow B_t d_{it}^2$
  - (iii) For each cluster  $t$  calculate the membership  $u_{it}$  of each data pattern  $x_i$  to the cluster where,

$$u_{it} \leftarrow \frac{1/B_{it}}{\sum_{r \in T_i} 1/B_{ir}} - \frac{1}{B_{it}} \left[ \frac{\sum_{r \in T_i} A_r / B_{ir}}{\sum_{r \in T_i} 1/B_{ir}} - A_t \right], t \in T_i$$

$$u_{it} \leftarrow 0, t \notin T_i$$

- (iv) If any  $u_{it} > 0$ , add  $t$  to set  $T_i$ , and repeat step 3 while some memberships are strictly negative.

4. For each cluster  $t$  update the cluster centre  $\mu_t$  and the scatter matrix  $\hat{S}_t$  as follows,

$$\mu_t \leftarrow \frac{\sum_i u_{it}^2 \mathbf{x}_i}{\sum_i u_{it}^2} \quad \hat{S}_t \leftarrow \frac{\sum_i u_{it}^2 (\mathbf{x}_i - \mu_t)(\mathbf{x}_i - \mu_t)^T}{\sum_i u_{it}}$$

5. If no membership  $u_{it}$  changed by more than a small value  $\varepsilon$  during the previous iteration, stop; otherwise return to step 2

Here the value of  $\varepsilon$  indicating convergence was set *a priori* to be 0.01.

## **4.2.1 Results**

### *4.2.1.1 Non-Overlapping Data*

#### *4.2.1.1.1 Six Clusters*

Data was partitioned each time so that in 5 out of 6 clusters, more than one species was in each cluster. The 6th cluster contained data for *M. pusilla* with negligible amounts of other species data, e.g. 0.8% or 0.4% which is not visible on the results charts. Three out of five times, one cluster contained data from many species. More than 90% of the data representing single species was grouped into one cluster for 8 out of 10 species. With 9 out of 10 species containing more than 80% of the corresponding data in a single cluster. In all cases, data representing *C. appendiculata*, *P. pupureum*, and *Rhodomonas sp.* were partitioned into the same cluster, these species are from different classes. In 3 out of 5 cases, *R. maculata*, *N. pyriformis*, and *H. brunnescens* were partitioned into the same cluster, these species were also from different classes. Finally, in 3 out of 5 applications, data representing *C. chiton* and *T. tetrathele* were grouped into the same cluster, and these species were also from different classes. (See Figure 4.1)

#### *4.2.1.1.2 Ten Clusters*

Four times, the data was partitioned into less than ten clusters and on two occasions, the data from many species was grouped into one large cluster. Looking at the results from all five applications, more than 90% of the data representing single species was grouped into one cluster for between 6 and 9 out

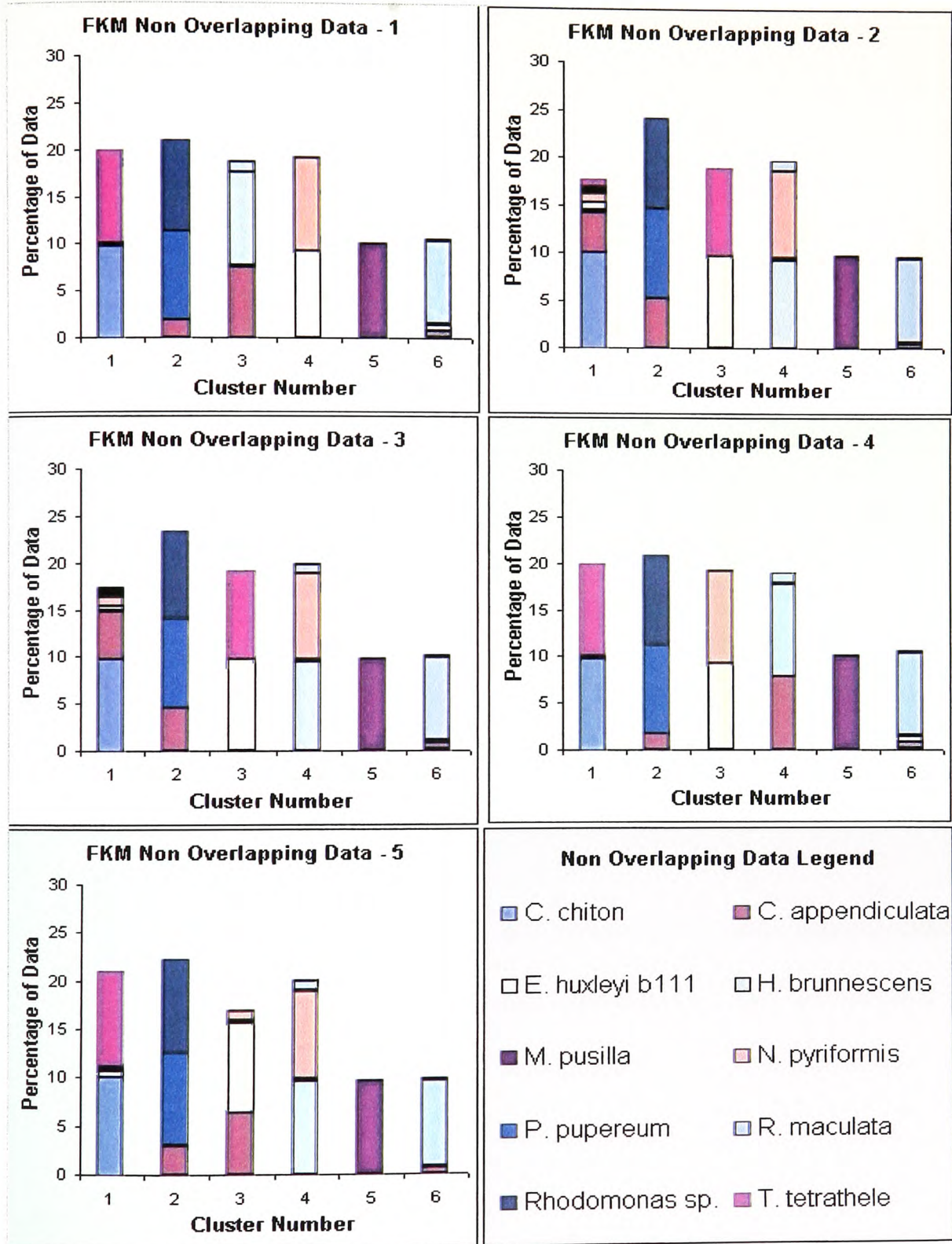
of 10 species. With between 7 and 10 out of 10 species containing more than 80% of the corresponding data in a single cluster. In the results for application number 5, cluster 1 actually contains 58% of the data, but the scale only goes up to 30% so the rest of the chart can be seen clearly. In 4 out of 5 of the applications, the data for *P. pupureum* and *Rhodomonas sp.* were grouped together, these two species are in different classes. In 3 out of 5 cases, the data for *N. pyriformis* and *M. pusilla* were grouped together, these two species are in the same class. (See Figure 4.2)

#### 4.2.1.1.3 Fourteen Clusters

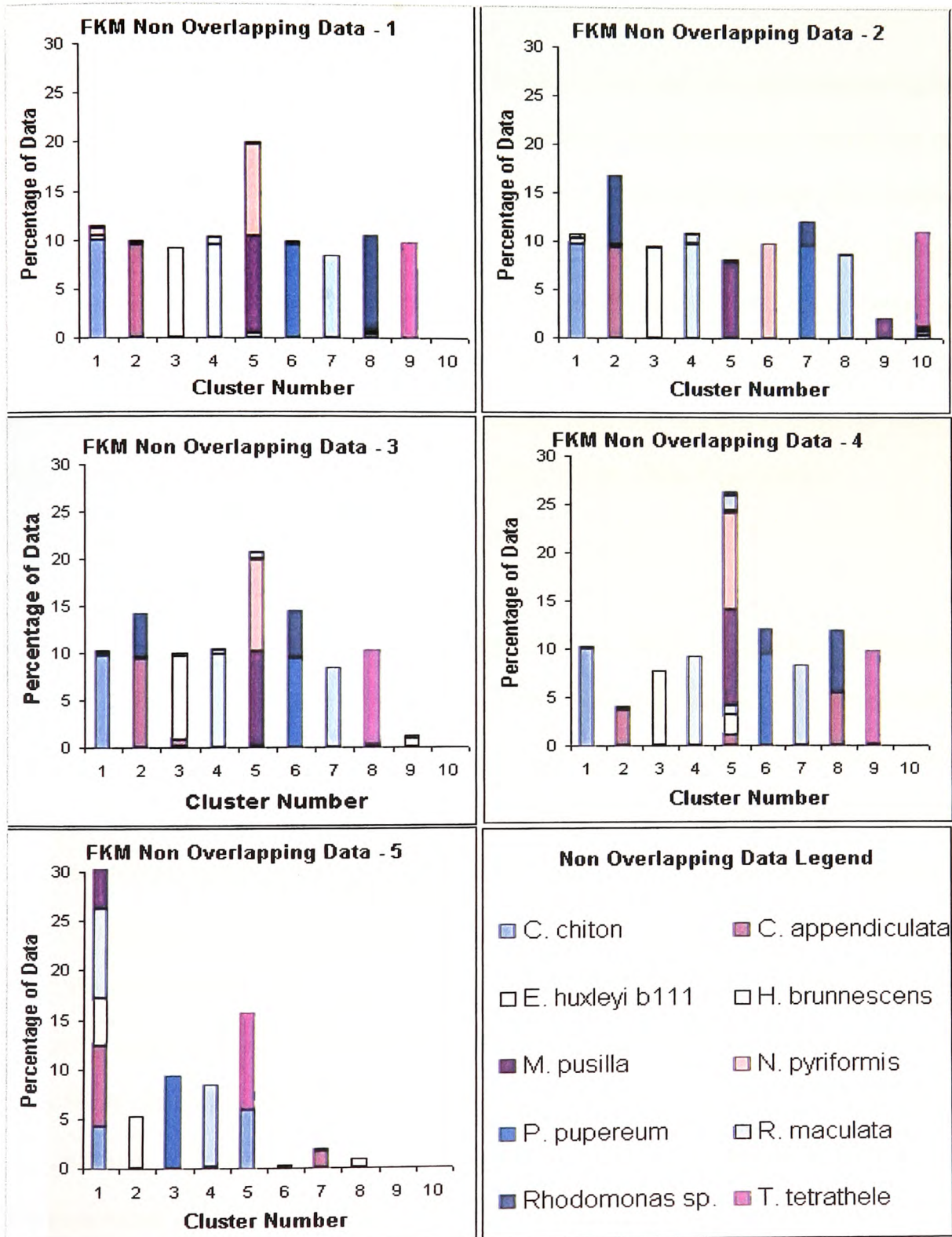
The algorithm did not converge when fourteen clusters were used to partition the data. It appears that this algorithm when applied to the non-overlapping data set, generates clustering solutions where the difference between two successive iterations of the algorithm, is greater than the specified value used to indicate convergence.



**Figure 4.1** FKM Non-Overlapping Data - Six Clusters



**Figure 4.2** FKM Non-Overlapping Data -Ten Clusters



#### 4.2.1.2 Overlapping Data

##### 4.2.1.2.1 Six Clusters

More than 90% of the data representing single species was grouped into one cluster for between 6 and 8 out of 10 species. With between 7 and 9 out of 10 species containing more than 80% of the corresponding data in a single cluster. In 4 out of 5 cases, data representing *C. chiton*, *A. pigmentosum*, and *A. coffaeiformis*, were grouped together where the species are from different classes, but which all belong to the group taxon. In all 5 cases, data representing *A. carterae*, *A. pigmentosum*, *G. veneficum* and *H. triquetra* were grouped together, where all species belong to the same class. (See Figure 4.3)

##### 4.2.1.2.2 Ten Clusters

Between 6 and 8 clusters contained data for different species partitioned into the same cluster. More than 90% of the data representing single species was grouped into one cluster for between 5 and 7 out of 10 species. With between 7 and 9 out of 10 species containing more than 80% of the corresponding data in a single cluster. In all cases, data representing *A. carterae* and *A. pigmentosum* were partitioned into the same cluster, these species are from the same class. In 4 out of 5 cases, data representing *G. veneficum* and *H. triquetra* were grouped together as was data representing *A. carterae* and *G. veneficum*. Each pair of species is from the same class. In 4 out of 5 cases, data representing *A. pigmentosum* and *C. chiton* were grouped together. Although these species are from different classes, the data from these species was shown

to overlap in the 3D plot of the overlapping data set, the species represented in this overlapping cluster are known as the 'group taxon'. The data representing the group taxon was analysed in further tests, the results of which are included later in this section. (See Figure 4.4)

#### 4.2.1.2.3 Fourteen Clusters

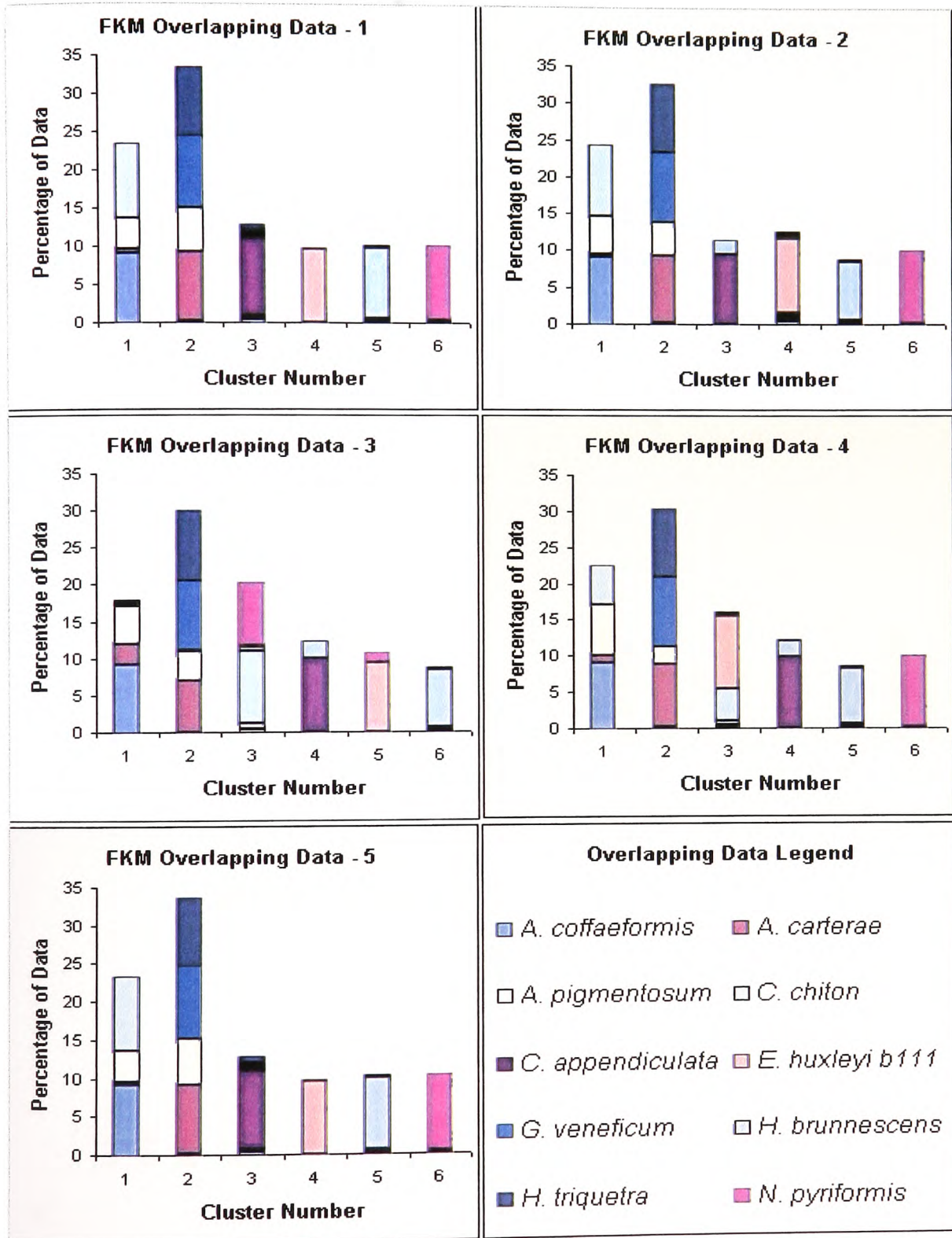
The data for a single species was partitioned over more clusters than with non-overlapping data, with some clusters containing negligible amounts of data. More than 90% of the data representing single species was grouped into one cluster for between 5 and 5 out of 10 species. With between 5 and 7 out of 10 species containing more than 80% of the corresponding data in a single cluster. In 4 out of 5 of the cases, 2 clusters contained very small amounts of data e.g. 0.29%. In all cases, data representing *A. carterae* and *A. pigmentosum* were grouped together as was data representing *A. carterae* and *G. veneficum*. Each pair of species is from the same class. In 3 out of 5 cases, data representing *G. veneficum* and *H. triquetra* were grouped together, both species also being in the same class. In 4 out of 5 cases, data representing *A. pigmentosum* and *C. chiton* were grouped together, which are from different classes but both belong to the group taxon. (See Figure 4.5)

#### 4.2.1.2.4 Five Clusters – One Taxon

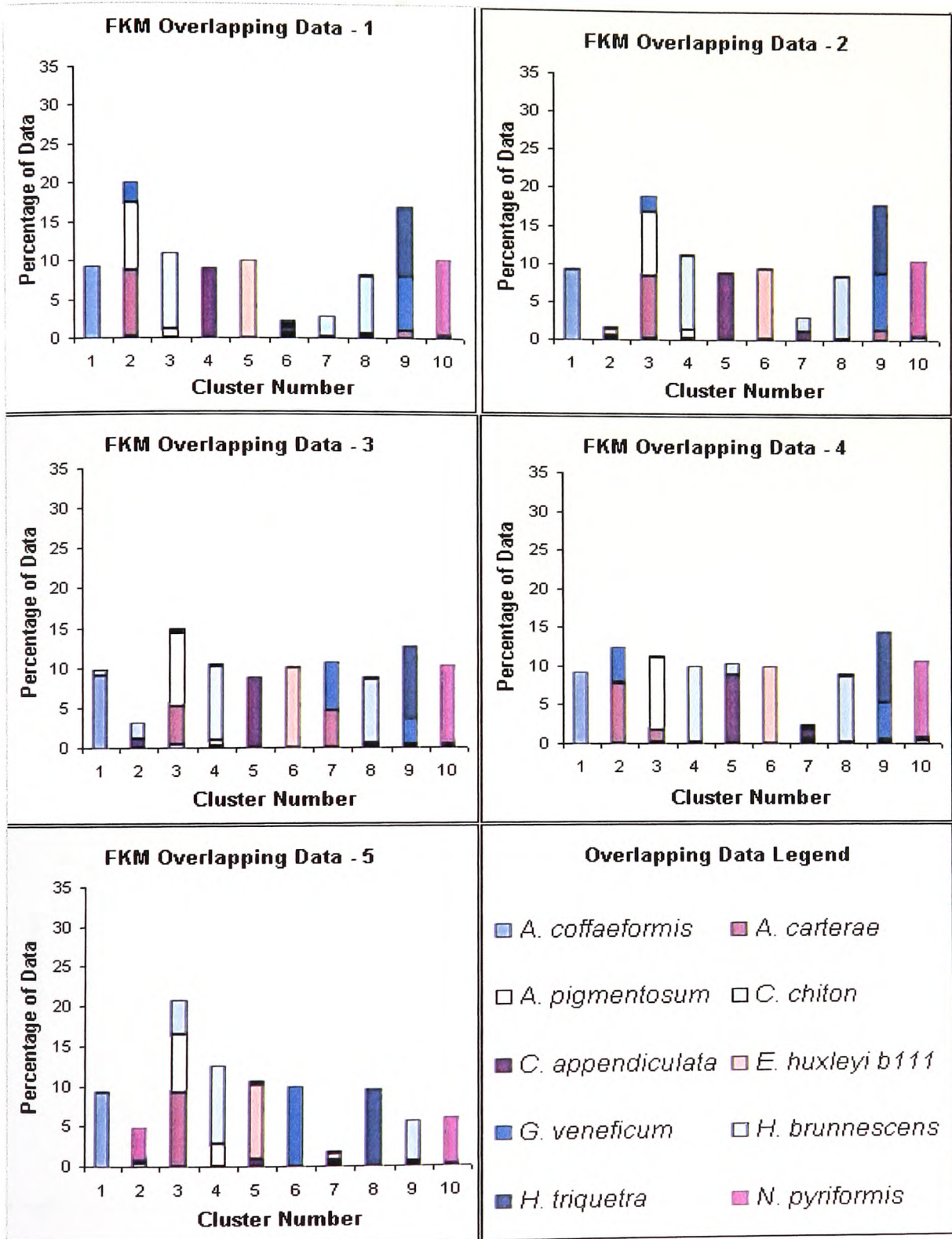
Between 97% and 98% of the taxon data was partitioned into 1 to 3 main clusters out of the five. In 4 out of 5 of the results, the main taxon clusters only contained group taxon data. In one case, in one cluster the taxon data was partitioned with data representing *E. huxleyi*, *C. appendiculata*, and *H. brunnescens* which are all in different classes to that of the species represented by the group taxon. In 4 out of 5 cases, data representing *E. huxleyi* and *N. pyriformis* were partitioned in the same cluster, they are in different classes. (See Figure 4.6)



**Figure 4.3** FKM Overlapping Data - Six Clusters

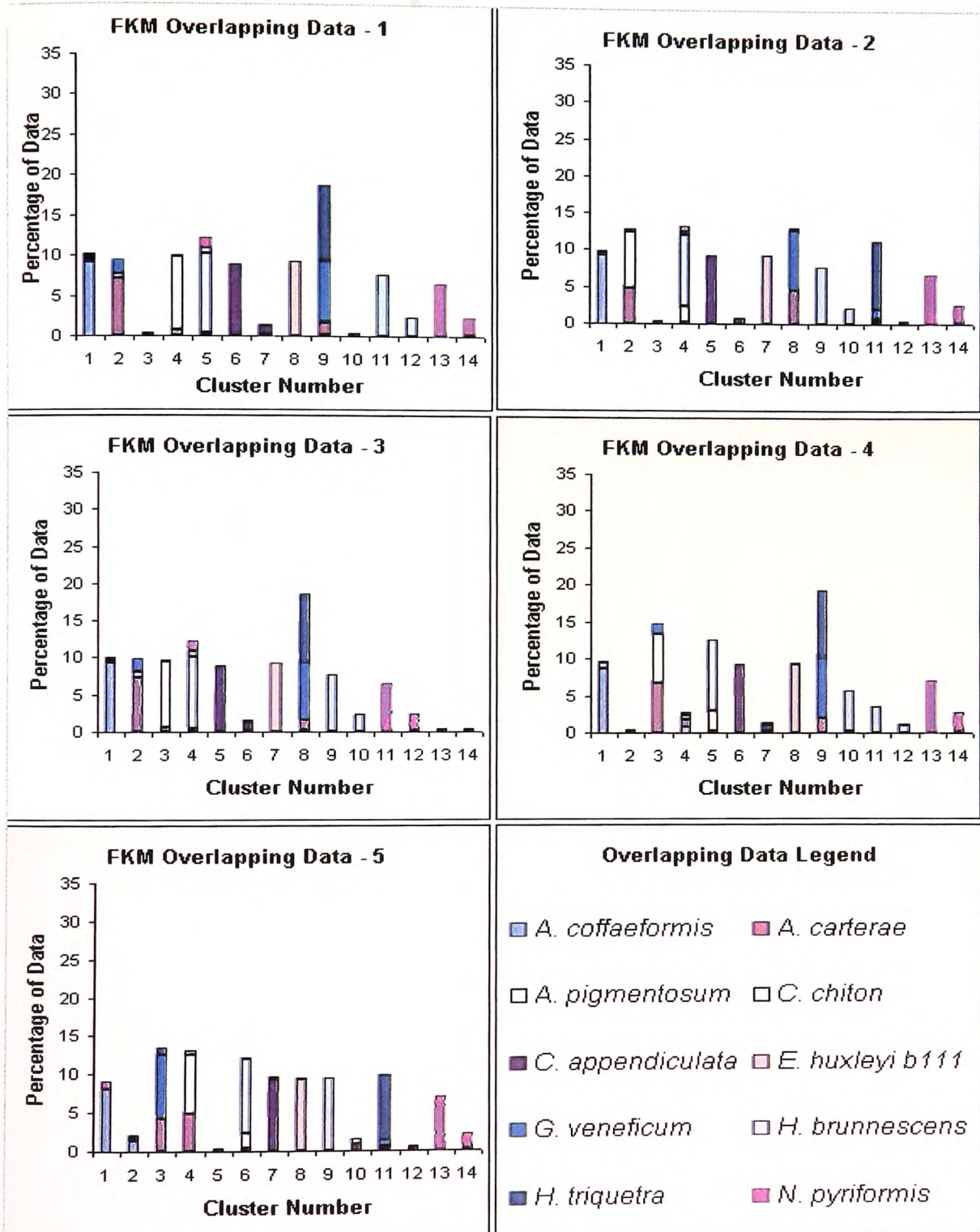


**Figure 4.4** FKM Overlapping Data - Ten Clusters



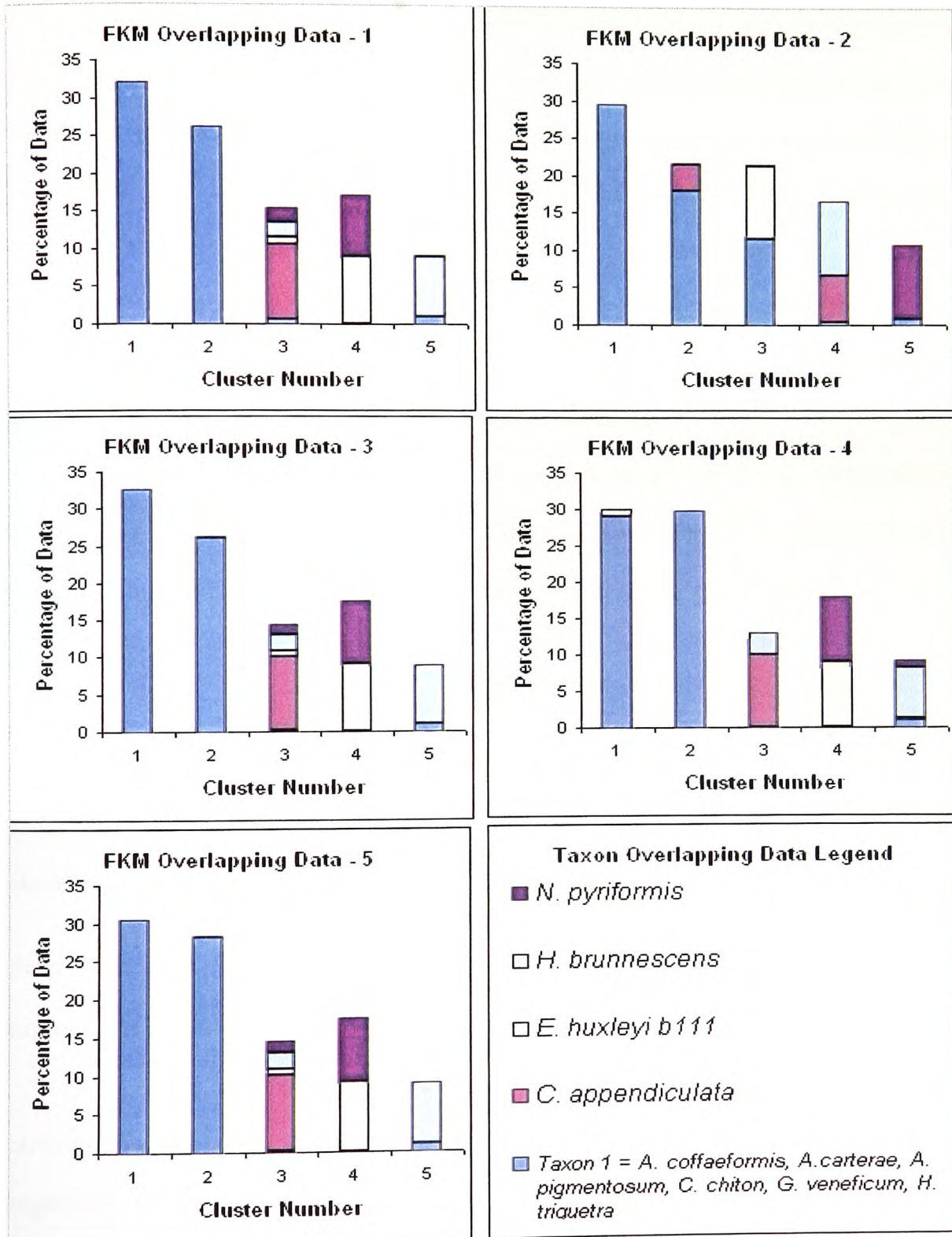


**Figure 4.5** FKM Overlapping Data - Fourteen Clusters





**Figure 4.6** FKM Overlapping Data - Five Clusters / One Taxon



### 4.3 Adaptive Distances AD

The AD algorithm, also known as the Gustafson-Kessel algorithm seeks to minimise the fuzzy sum of squared generalised distances of the data patterns to the cluster centres [36]. This is subject to the constraint that the determinant of the scatter matrices (a measure of cluster volume) can be fixed in advance. However, bad clustering can result if unsuitable values are chosen.

In order for this technique to be successful, the data should be divided into groups of similar numbers of elements so that all the data can be allocated. Also the dispersion within each cluster should be minimised.

This algorithm follows the notation in the paper by P. J. Rousseeuw published in 1996 and is obtained by changing the manner of calculating the quantities  $A_i$  and  $B_{it}$  which are variables used to represent the objective function, so that [36] :

$$A_i = \frac{1}{2} \frac{\tau}{\beta} n_i^{p\beta-\tau-1} (\theta_i |\hat{S}_i|)^\beta,$$

$$B_{it} = n_i^{p\beta-e-1} (\theta_{it} |\hat{S}_i|)^\beta$$

where  $p$  is the data dimensionality,  $n_i = \sum_i u_{ii}$ , and the parameters  $\beta$  and  $\tau$  are given by  $\beta = 1/p$ , and  $\tau = 0$  where  $\beta$  and  $\tau$  are variables used in order that the formula can be generalised.

The focus of this method is to partition the data in clusters of similar cardinality, a low level of dispersion, and equal volume. This is achieved by using equations  $A_i$  and  $B_{it}$  above which adapts the general algorithm to the AD approach.

The Adaptive Distances algorithm used is as follows:

1. For each cluster  $t$  initialise the cluster centre  $\mu_t$  to a randomly-selected data pattern and initialise the scatter matrix  $\hat{S}_t$  to the identity matrix
2. For each cluster  $t$ , calculate  $A_t$  and  $B_{it}$  which are variables used to simplify the relevant objective function where,

$$A_t = \frac{1}{2} \frac{\tau}{\beta} n_t^{p\beta-\tau-1} (\theta_t |\hat{S}_t|)^\beta \text{ and}$$

$$B_{it} = n_t^{p\beta-e-1} (\theta_t |\hat{S}_t|)^\beta$$

3. Calculate memberships: for each data pattern  $x_i$ :

- (i) Initialise  $T_i$  to an empty set where  $T_i$  is a set of indices  $t$  of the clusters that data pattern  $x_i$  belongs to,
- (ii) For each cluster  $t$  calculate the distance from each data pattern  $x_i$  to the cluster centre  $B_{it} \leftarrow B_t d_{it}^2$
- (iii) For each cluster  $t$  calculate the membership  $u_{it}$  of each data pattern  $x_i$  to the cluster where,

$$u_{it} \leftarrow \frac{1/B_{it}}{\sum_{r \in T_i} 1/B_{ir}} - \frac{1}{B_{it}} \left[ \frac{\sum_{r \in T_i} A_r / B_{ir}}{\sum_{r \in T_i} 1/B_{ir}} - A_t \right], t \in T_i$$

$$u_{it} \leftarrow 0, t \notin T_i$$

- (iv) If any  $u_{it} > 0$ , add  $t$  to set  $T_i$ , and repeat step 3 while some memberships are strictly negative.

4. For each cluster  $t$  update the cluster centre  $\mu_t$  and the scatter matrix  $\hat{S}_t$  as follows,

$$\mu_t \leftarrow \frac{\sum_i u_{it}^2 \mathbf{x}_i}{\sum_i u_{it}^2} \quad \hat{S}_t \leftarrow \frac{\sum_i u_{it}^2 (\mathbf{x}_i - \mu_t)(\mathbf{x}_i - \mu_t)^T}{\sum_i u_{it}^2}$$

5. If no membership  $u_{it}$  changed by more than a small value  $\varepsilon$  during the previous iteration, stop; otherwise return to step 2.

Here, a reasonable *a priori* assumption is that all clusters should start with the same volume; i.e.  $\theta_t = 1$  for all  $t$  where  $\theta_t$  is the constraint of unknown positive definite matrix  $G_t$  used to avoid the determinant of  $G_t$  becoming zero.  $G_t$  is an unknown positive definite matrix used in the original objective function for this algorithm, and the value of  $\varepsilon$  indicating convergence was taken to be 0.01.

### **4.3.1 Results**

#### *4.3.1.1 Non-Overlapping Data*

##### *4.3.1.1.1 Six Clusters*

This algorithm partitioned the data in six clusters on each application. More than 90% of the data representing single species was grouped into one cluster for between 7 and 8 out of 10 species. With between 8 and 10 out of 10 species containing more than 80% of the corresponding data in a single cluster. In all 5 cases, data representing *H. brunnescens* and *R. maculata*, *C. appendiculata* and *Rhodomonas sp.*, and *E. huxleyi* and *N. pyriformis* were grouped together in the relevant pairs. *H. brunnescens* and *R. maculata* are from the same class, whereas *C. appendiculata* and *Rhodomonas sp.*, and *E. huxleyi* and *N. pyriformis* are from different classes. In 4 out of 5 cases, data representing *C. chiton* and *T. tetrathele* were grouped together, these species are also from different classes. (See Figure 4.7)

##### *4.3.1.1.2 Ten Clusters*

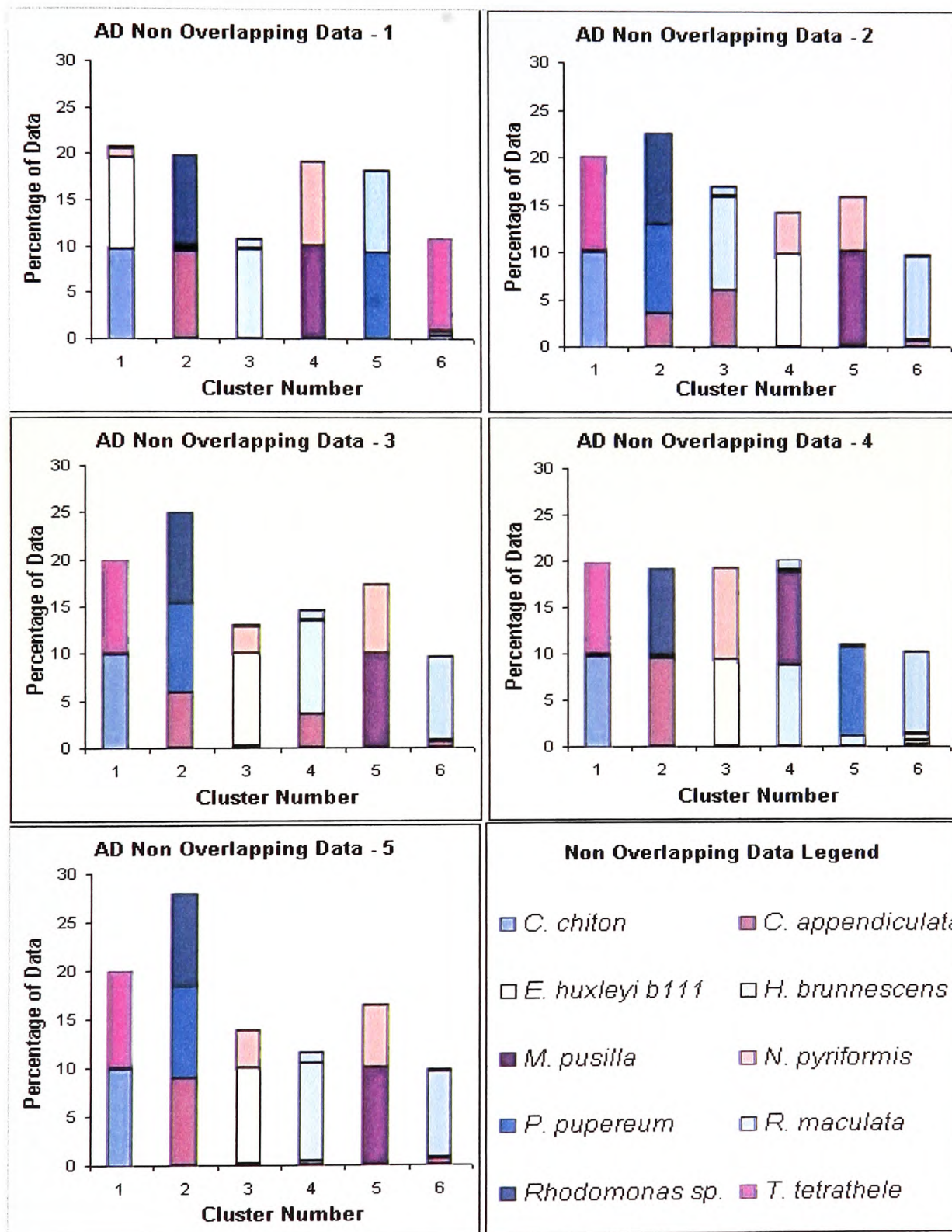
This algorithm partitioned the data in ten clusters on each application. More than 90% of the data representing single species was grouped into one cluster for between 6 and 7 out of 10 species. With between 8 and 10 out of 10 species containing more than 80% of the corresponding data in a single cluster. In all 5 cases, data representing *H. brunnescens* and *R. maculata*, were grouped together, these species are from different classes. Also in all 5 cases, data representing *M. pusilla*, and *N. pyriformis* were grouped together, these species

belong to the same class. In 3 out of the 5 cases, data representing *C. appendiculata* and *H. brunnescens* were grouped together, these species also belong to the same class. (See Figure 4.8)

#### 4.3.1.1.3 Fourteen Clusters

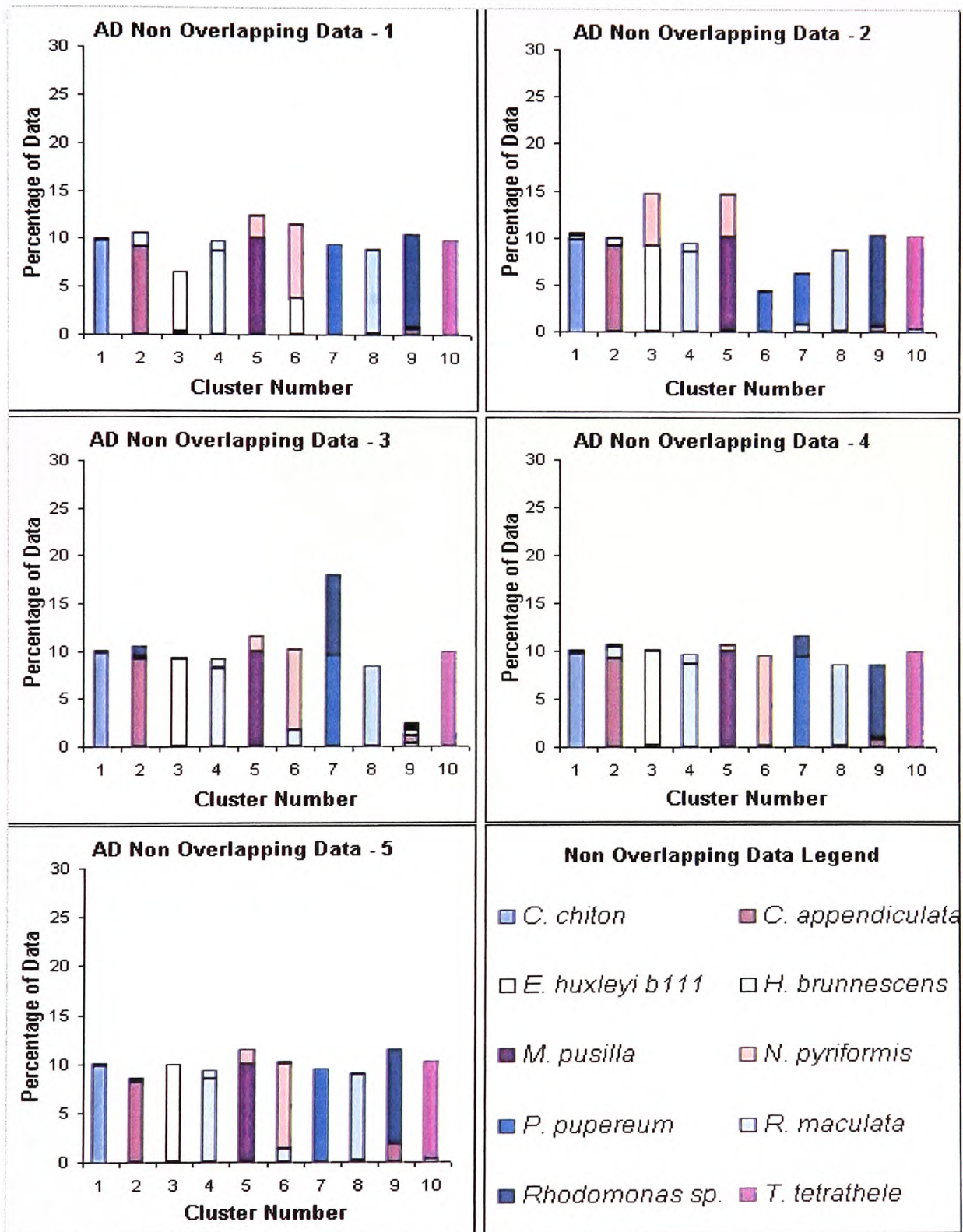
This algorithm partitioned the data into fourteen clusters on each application. More than 90% of the data representing single species was grouped into one cluster for between 5 and 6 out of 10 species. With between 8 and 9 out of 10 species containing more than 80% of the corresponding data in a single cluster. In all 5 cases, data representing *H. brunnescens* and *R. maculata*, were grouped together, these species are from different classes. (See Figure 4.9)

**Figure 4.7** AD Non-Overlapping - Six Clusters



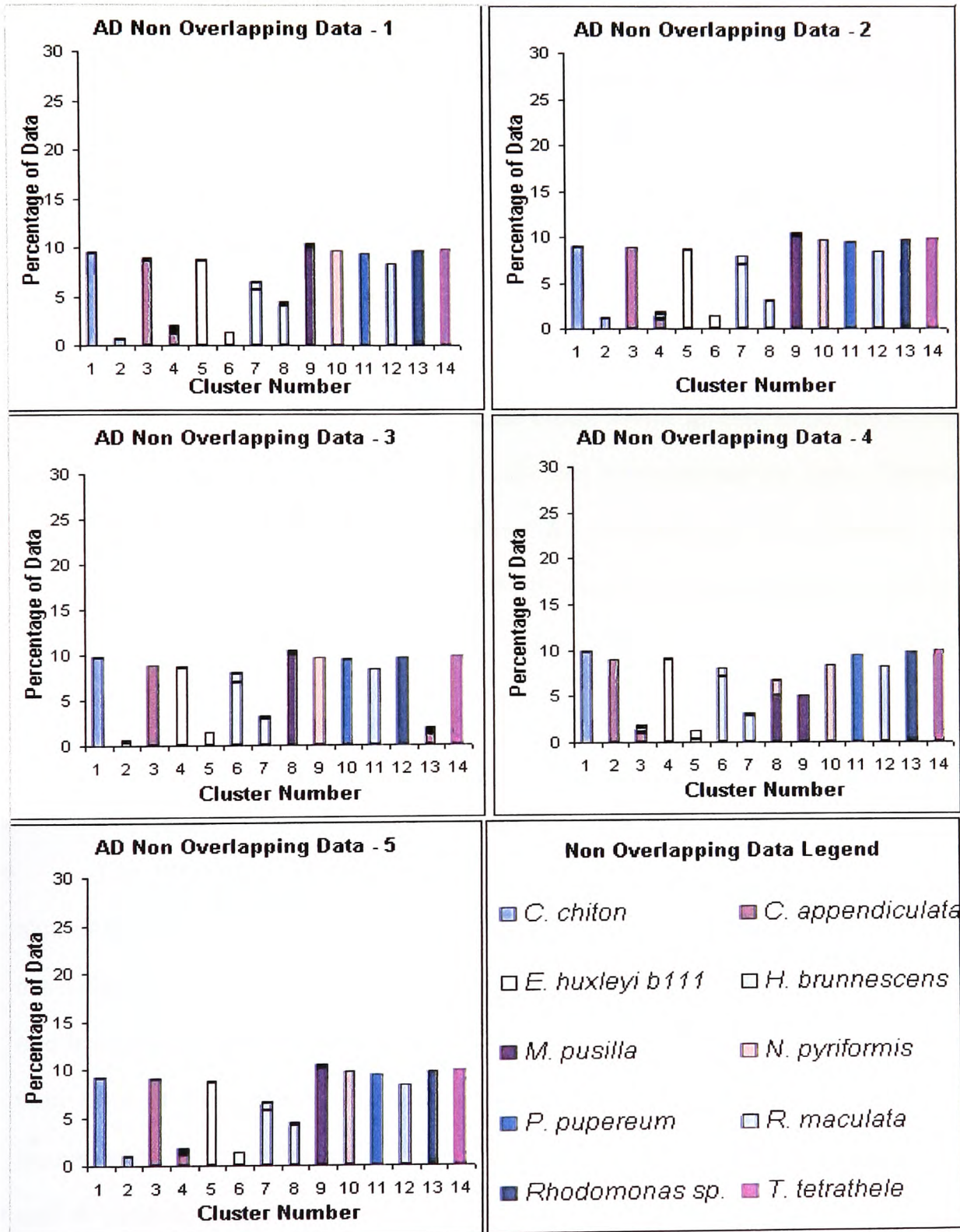


**Figure 4.8** AD Non-Overlapping Data - Ten Clusters





**Figure 4.9 AD Non-Overlapping Data - Fourteen Clusters**



#### 4.3.1.2 Overlapping Data

##### 4.3.1.2.1 Six Clusters

There were some similarities between the results of the five applications of this algorithm. More than 90% of the data representing single species was grouped into one cluster for between 6 and 9 out of 10 species. With between 7 and 10 out of 10 species containing more than 80% of the corresponding data in a single cluster. In all 5 cases, data representing *G. veneficum* and *H. triquetra*, and *A. carterae* and *G. veneficum* were grouped together in the relevant pairings, pair-wise these species are from the same class. There tended to be more data split into two significant clusters than with the non-overlapping data. Various combinations of the data representing *A. coffaeformis*, *A. carterae*, *A. pigmentosum*, *C. chiton*, *G. veneficum* and *H. triquetra* were partitioned together. Although these species are in different classes, they belong to the group taxon. (See Figure 4.10)

##### 4.3.1.2.2 Ten Clusters

The data was grouped in a similar fashion for each application of the algorithm and partitioned the data into ten clusters each time. More than 90% of the data representing single species was grouped into one cluster for between 4 and 6 out of 10 species. With between 7 and 8 out of 10 species containing more than 80% of the corresponding data in a single cluster. In all 5 cases, data representing *A. carterae* and *A. pigmentosum*, *G. veneficum* and *H. triquetra*, and *A. carterae* and *G. veneficum* were grouped together in the relevant pairings,

pair-wise these species are from the same class. In 4 out of 5 cases, data representing *C. appendiculata* and *H. brunnescens* were partitioned together, these species are also in the same class. (See Figure 4.11)

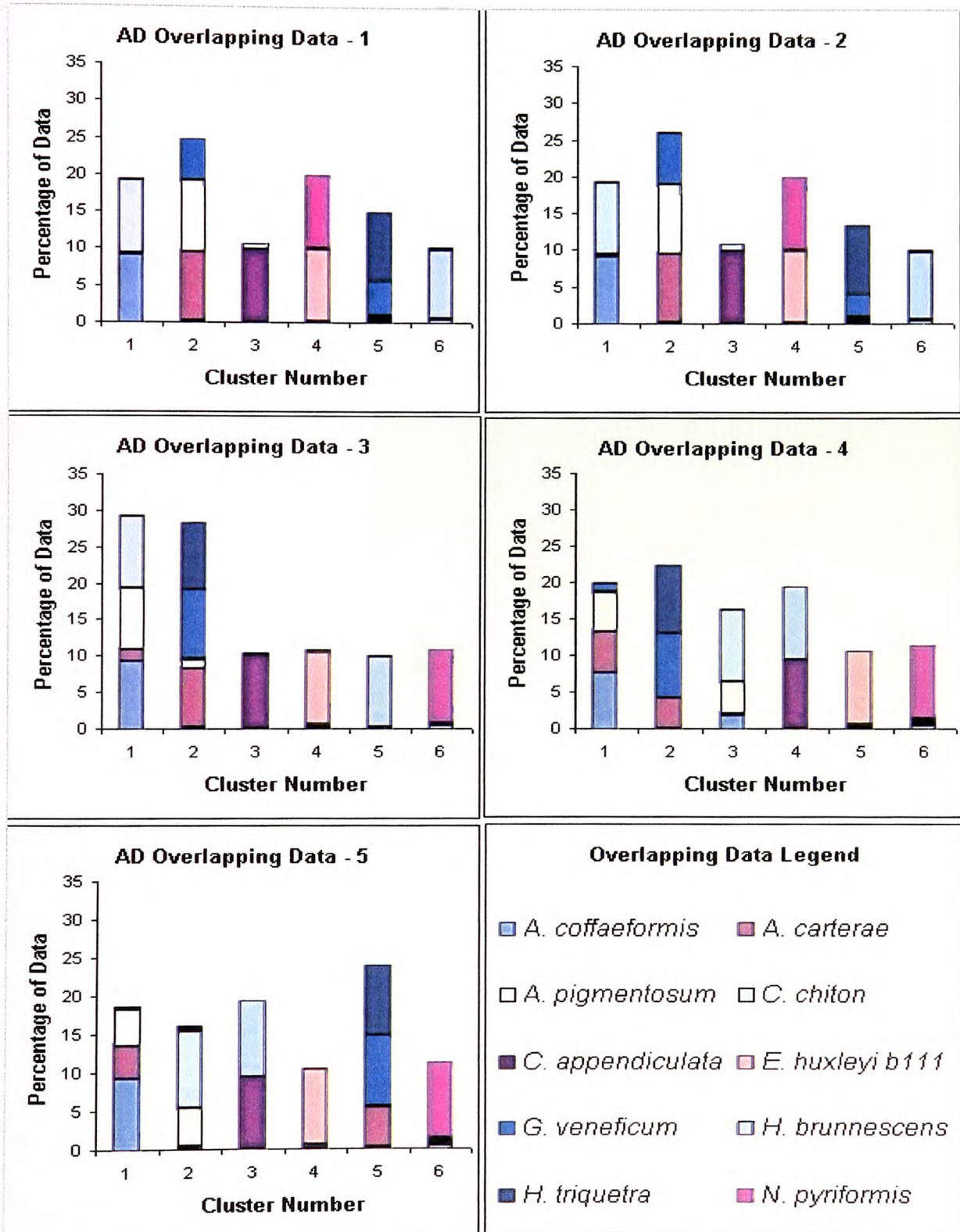
#### 4.3.1.2.3 Fourteen Clusters

There were also several similarities between the results of each application of this algorithm, and the data was partitioned into fourteen clusters each time. More than 90% of the data representing single species was grouped into one cluster for between 2 and 4 out of 10 species. With between 4 and 6 out of 10 species containing more than 80% of the corresponding data in a single cluster. In all 5 cases, data representing *A. carterae* and *A. pigmentosum*, *G. veneficum* and *H. triquetra*, and *A. carterae* and *G. veneficum* were grouped together in the relevant pairings, pair-wise these species are from the same class. (See Figure 4.12)

#### 4.3.1.2.4 Five Clusters – One Taxon

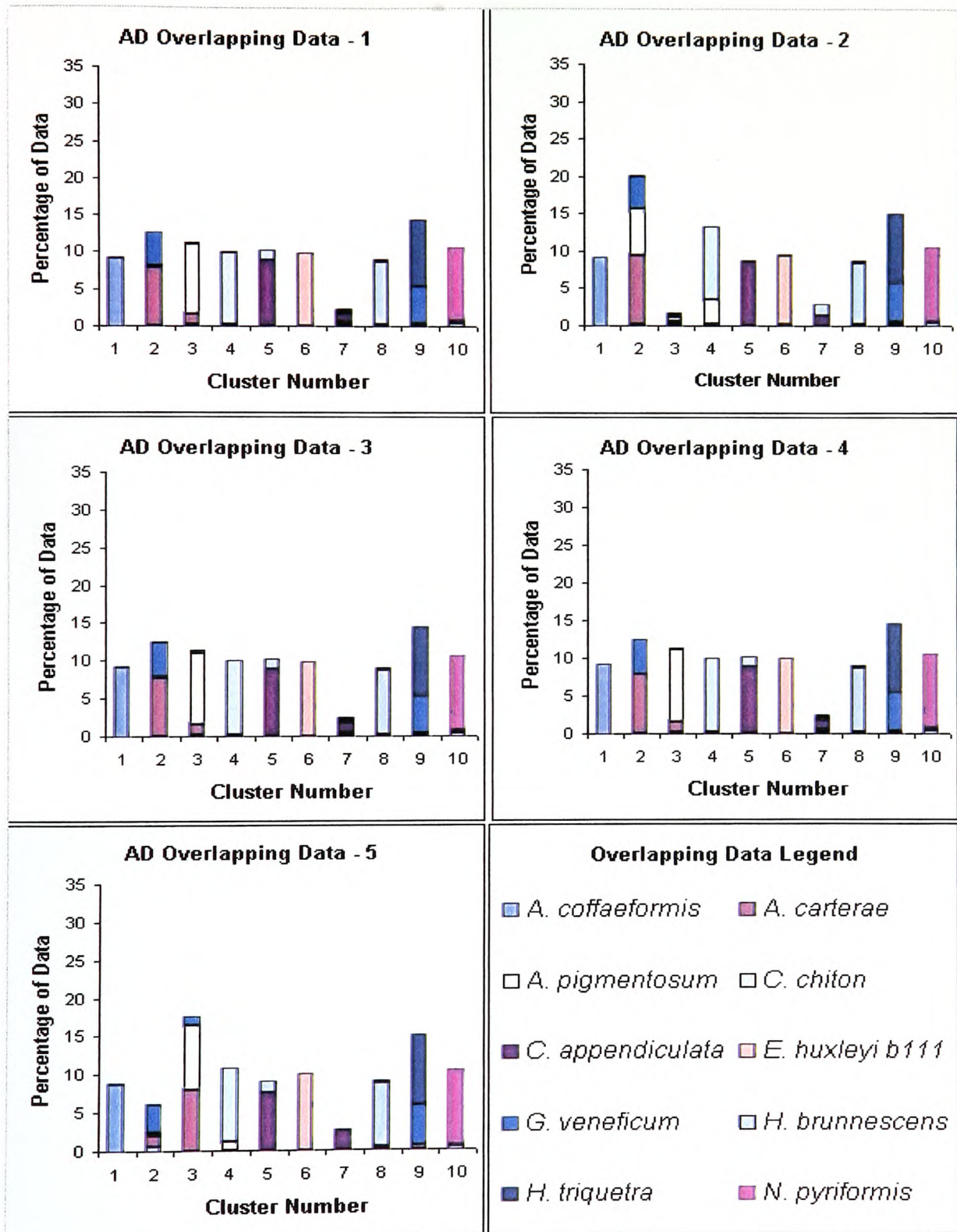
Between 95% and 98% of the taxon data was partitioned into 2 main clusters out of the five. In 4 out of 5 cases, data representing *C. appendiculata* and *E. huxleyi* were partitioned in the same cluster, these species are in different classes. In 3 out of 5 cases, data representing *H. Brunnescens* and *N. pyriformis* were partitioned in the same cluster, they are also in different classes. The data from the other taxons were grouped together in clusters in a similar manner for each application. (See Figure 4.13)

**Figure 4.10** AD Overlapping Data - Six Clusters

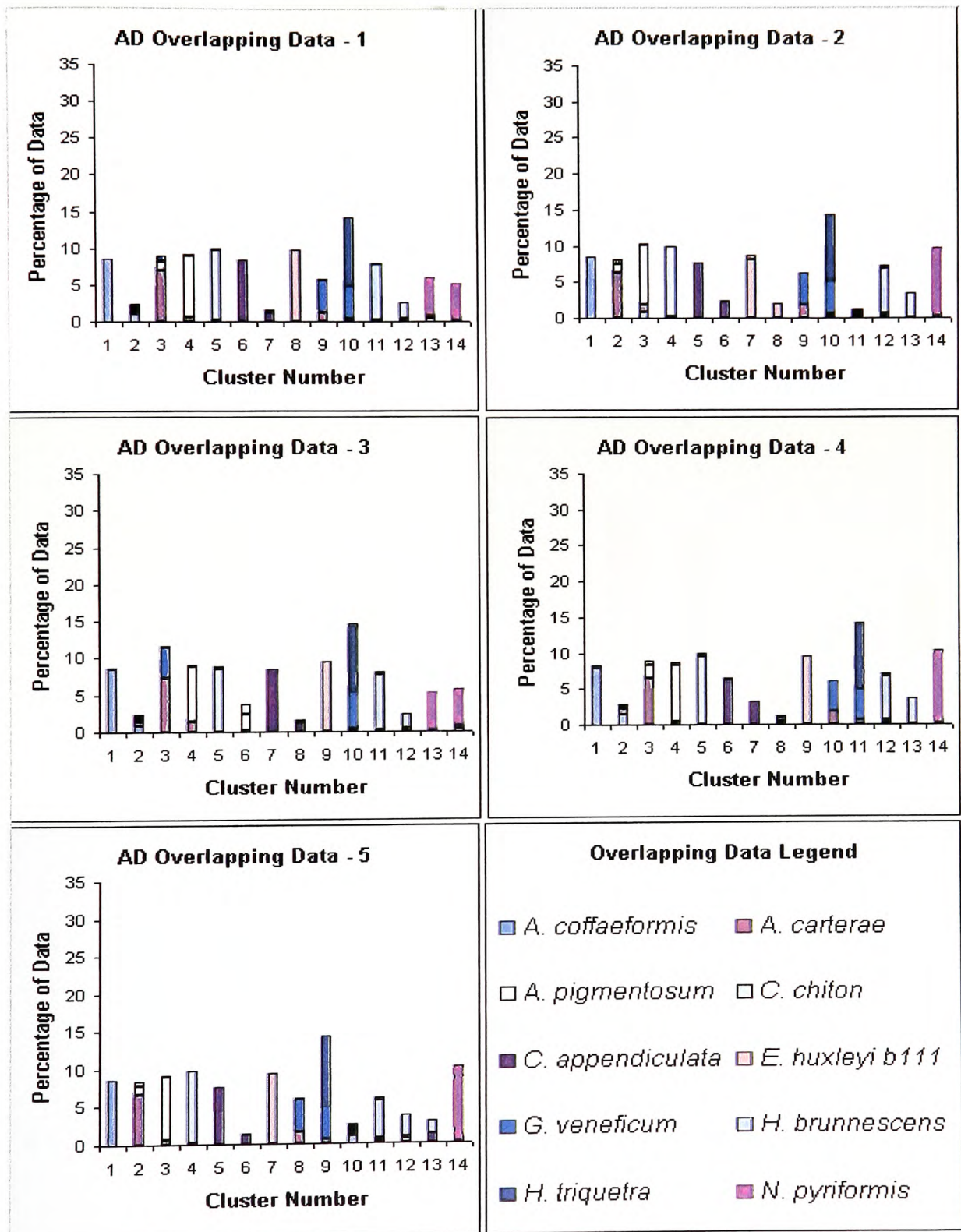




**Figure 4.11 AD Overlapping Data - Ten clusters**

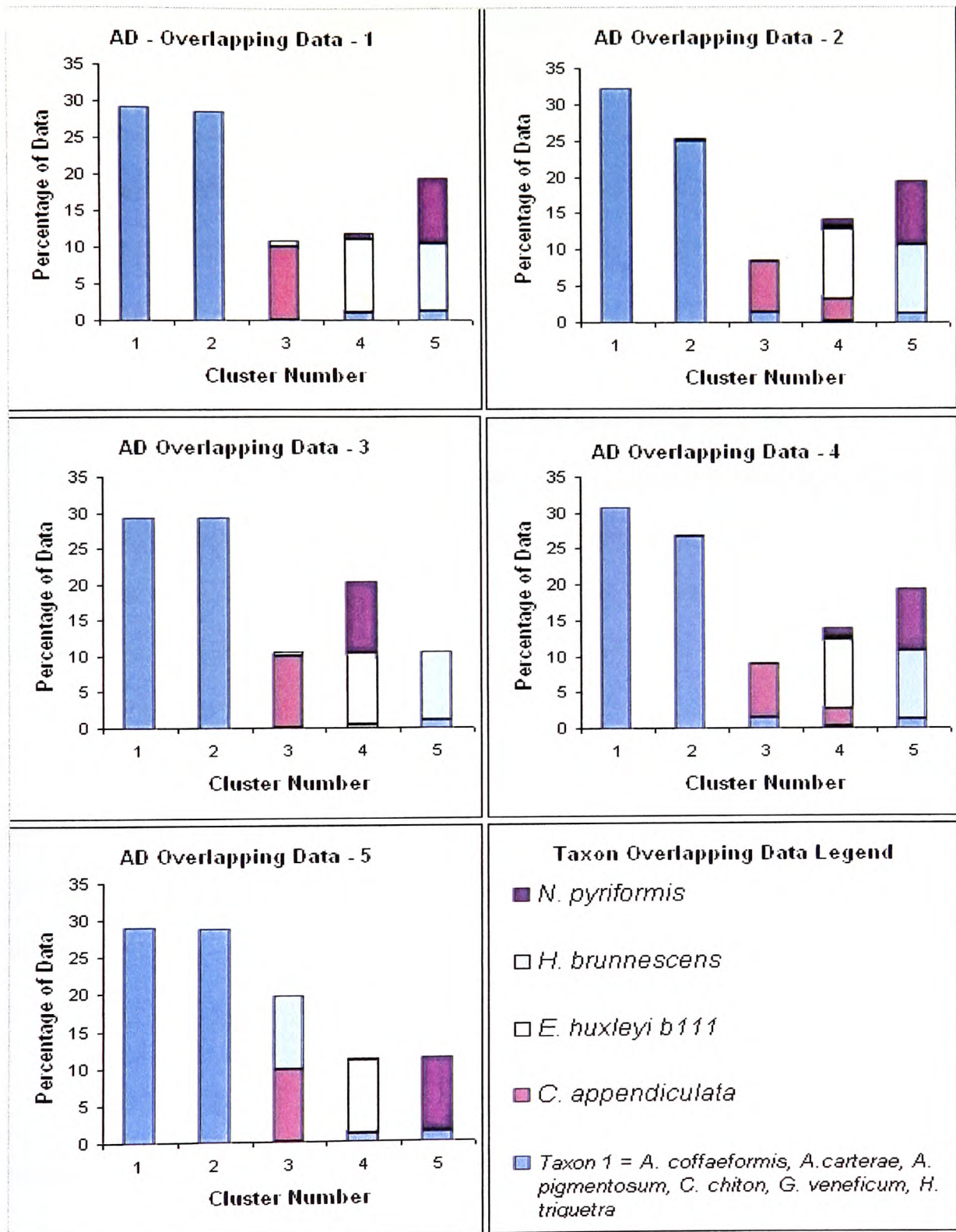


**Figure 4.12 AD Overlapping Data - Fourteen Clusters**





**Figure 4.13 AD Overlapping Data – Five Clusters / One Taxon**



#### 4.4 Maximum Likelihood ML

The ML algorithm assumes that each cluster represents a multivariate normal probability distribution, and attempts to find a clustering solution that maximises the overall likelihood of the data set over all clusters and data items; this method tends to seek cluster solutions where all clusters have similar volumes [36]. In this case,  $\mu_t$  and  $S_t$  are unknown for  $t$ , and  $i_t$  is the “set of indices of the observations belonging to cluster  $t$ ” [50]. The likelihood function can be inferred as the product of the objective functions for all  $i$  [51]. Probable values of  $i_t$ ,  $\mu_t$  and  $S_t$  can be calculated by maximising this likelihood function.

This approach not only has the ability to recognise ellipse shaped clusters which have different orientations, but also has the property that no constants have to be set *a priori*, unlike Adaptive Distances. Unfortunately a bias occurs towards results with clusters of similar volumes, when the clusters actually have wide ranging volumes.

This algorithm follows the notation in the paper by P. J. Rousseeuw published in 1996 [36], and is obtained by changing the manner of calculating the quantities  $A_t$  and  $B_{it}$  which are variables used to represent the relevant objective function, so that:

$$A_t = -\frac{1}{2} \log |\hat{S}_t|,$$

$$B_{it} = (\mathbf{x}_i - \boldsymbol{\mu}_t)' (\hat{S}_t)^{-1} (\mathbf{x}_i - \boldsymbol{\mu}_t)$$

where  $\boldsymbol{\mu}_t$  is the cluster centre of cluster  $t$ ,  $\hat{S}_t$  is the scatter matrix of cluster  $t$ , and  $\mathbf{x}_i$  is the  $i$ th data point.



The focus of this method is to maximise the likelihood of producing a clustering solution which matches the natural clusters in the data. This is achieved by using equations  $A_t$  and  $B_{it}$  above which adapts the general algorithm to the ML approach. It assumes each data object is drawn from a multivariate normal distribution  $N(\boldsymbol{\mu}_t, \hat{\mathbf{S}}_t)$ .

The Maximum Likelihood algorithm used is as follows:

1. For each cluster  $t$  initialise the cluster centre  $\boldsymbol{\mu}_t$  to a randomly-selected data pattern and initialise the scatter matrix  $\hat{\mathbf{S}}_t$  to the identity matrix
2. For each cluster  $t$ , calculate  $A_t$  and  $B_{it}$  which are variables used to simplify the objective function where,

$$A_t = -\frac{1}{2} \log |\hat{\mathbf{S}}_t| \text{ and}$$

$$B_{it} = (\mathbf{x}_i - \boldsymbol{\mu}_t)' (\hat{\mathbf{S}}_t)^{-1} (\mathbf{x}_i - \boldsymbol{\mu}_t)$$

3. Calculate memberships: for each data pattern  $\mathbf{x}_i$ :
  - (i) Initialise  $T_i$  to an empty set where  $T_i$  is a set of indices  $t$  of the clusters that data pattern  $\mathbf{x}_i$  belongs to,
  - (ii) For each cluster  $t$  calculate the distance from each data pattern  $\mathbf{x}_i$  to the cluster centre  $B_{it} \leftarrow B_t d_{it}^2$
  - (iii) For each cluster  $t$  calculate the membership  $u_{it}$  of each data pattern  $\mathbf{x}_i$  to the cluster where,

$$u_{it} \leftarrow \frac{1/B_{it}}{\sum_{r \in T_i} 1/B_{ir}} - \frac{1}{B_{it}} \left[ \frac{\sum_{r \in T_i} A_r/B_{ir}}{\sum_{r \in T_i} 1/B_{ir}} - A_t \right], t \in T_i$$

$$u_{it} \leftarrow 0, t \notin T_i$$

- (iv) If any  $u_{it} > 0$ , add  $t$  to set  $T_i$ , and repeat step 3 while some memberships are strictly negative.

4. For each cluster  $t$  update the cluster centre  $\mu_t$  and the scatter matrix  $\hat{S}_t$  as follows,

$$\mu_t \leftarrow \frac{\sum_i u_{it}^2 \mathbf{x}_i}{\sum_i u_{it}^2} \quad \hat{S}_t \leftarrow \frac{\sum_i u_{it}^2 (\mathbf{x}_i - \mu_t)(\mathbf{x}_i - \mu_t)^T}{\sum_i u_{it}^2}$$

5. If no membership  $u_{it}$  changed by more than a small value  $\varepsilon$  during the previous iteration, stop; otherwise return to step 2.

Here the value of  $\varepsilon$  indicating convergence was set *a priori* to be 0.01

#### **4.4.1 Results**

##### *4.4.1.1 Non-Overlapping Data*

###### *4.4.1.1.1 Six Clusters*

Similarly as for ten and fourteen clusters, the data was partitioned so that between, 14% and 32% of the data was combined into one cluster. More than 90% of the data representing single species was grouped into one cluster for between 1 and 7 out of 10 species. With between 2 and 9 out of 10 species containing more than 80% of the corresponding data in a single cluster. In all applications the data representing *M. pusilla* and *N. pyriformis* were partitioned in the same cluster, these species are in the same class. In 4 out of 5 cases, data representing *G. veneficum* and *H. triquetra* were grouped together, these species are in different classes. In 3 out of 5 cases, data representing *C. chiton* and *T. tetrathele* was partitioned into the same cluster, these species belong to different classes. Also in 3 out of 5 cases, data representing *C. appendiculata* and *H. brunnescens* were grouped together and these species belong to the same class. (See Figure 4.14)

###### *4.4.1.1.2 Ten Clusters*

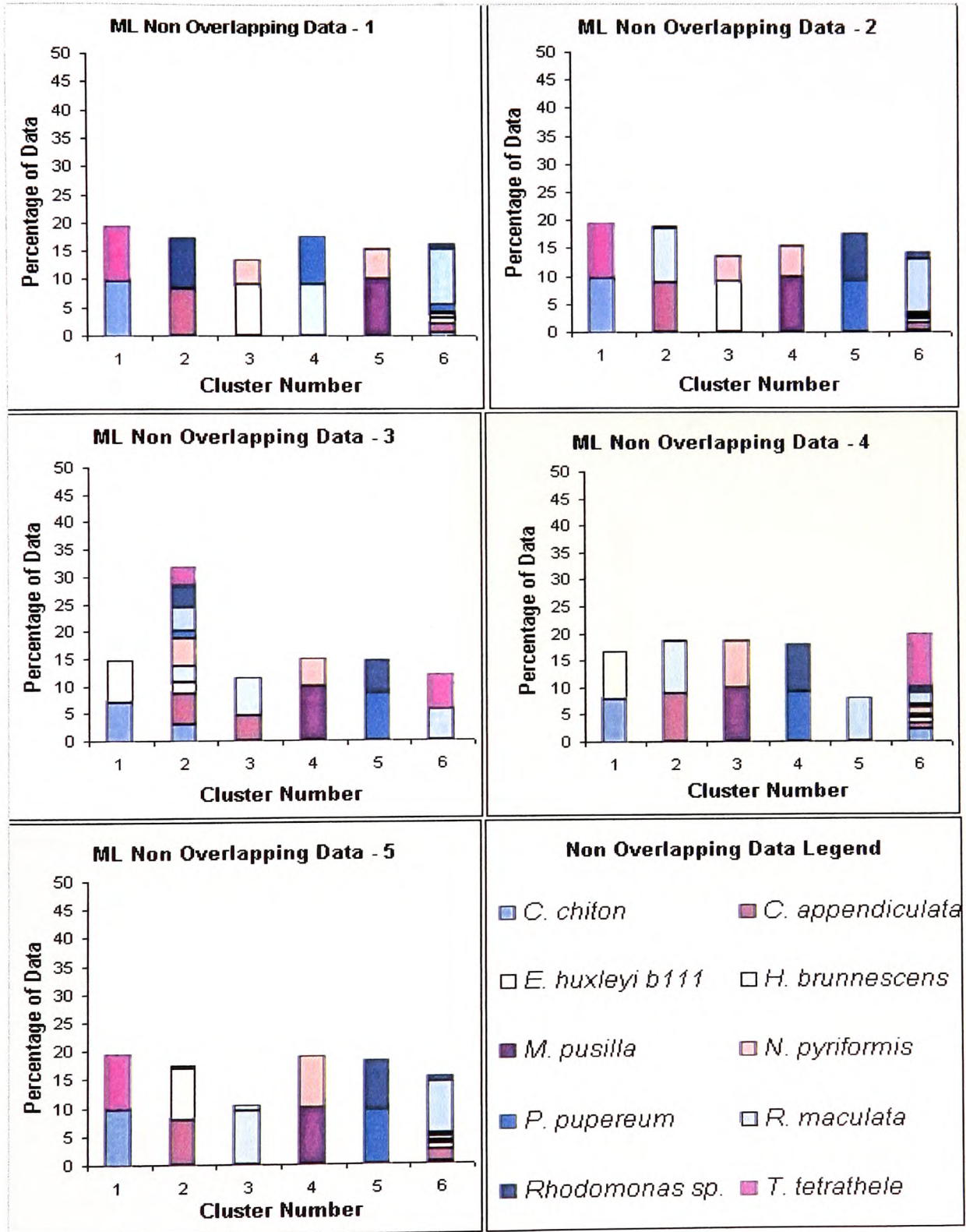
The algorithm partitioned the data into ten clusters on each application. However, it had a tendency to partition the data so that nine clusters contained data for a single species but with between 40% and 100% of the data for that species. The tenth cluster contained all the remaining data, meaning data from nine or ten species were present in this very large cluster, between 39% and

43% of all the data were contained in this single cluster. More than 90% of the data representing single species was grouped into one cluster for between 1 and 7 out of 10 species. With between 1 and 9 out of 10 species containing more than 80% of the corresponding data in a single cluster. (See Figure 4.15)

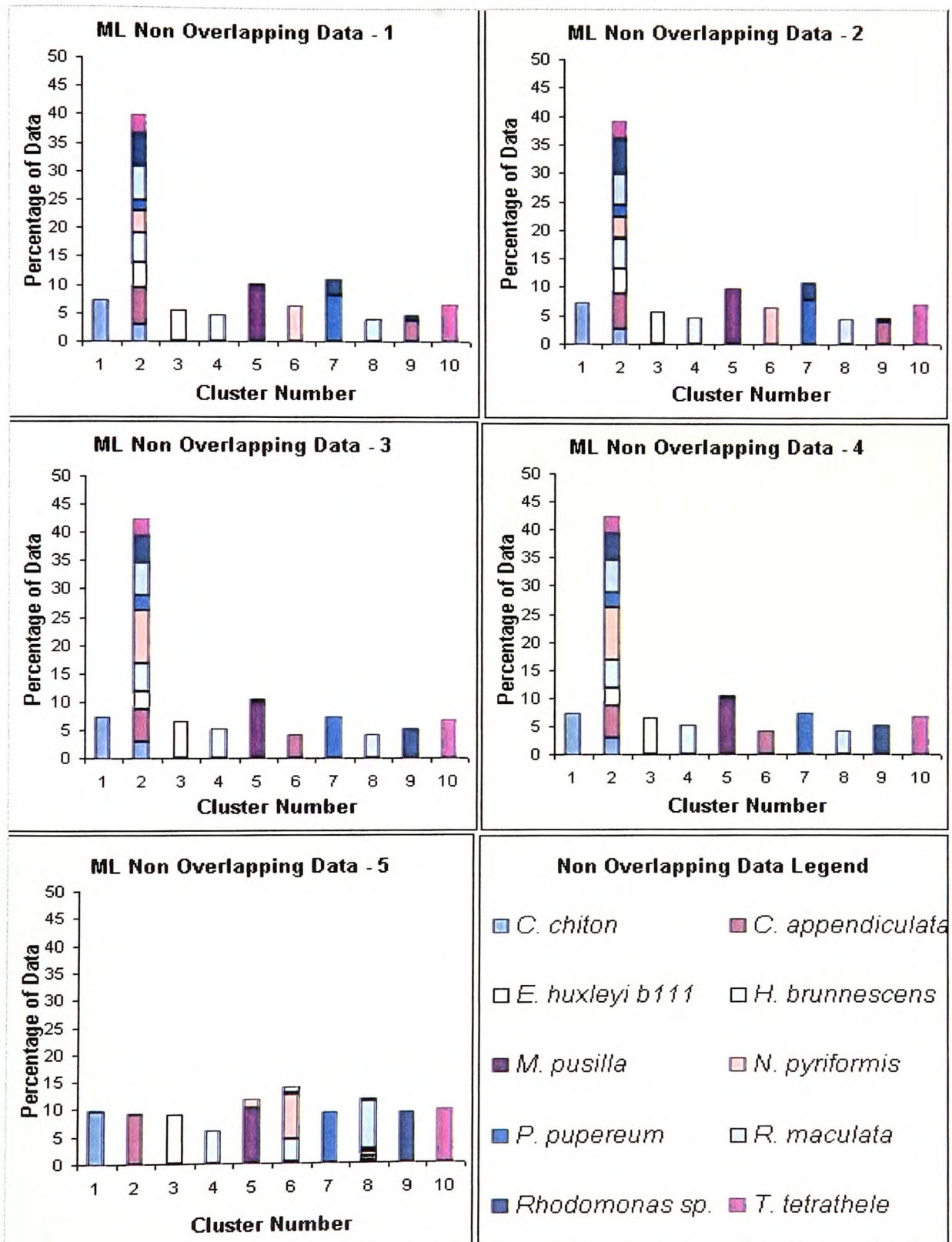
#### *4.4.1.1.3 Fourteen Clusters*

The algorithm partitioned the data into fourteen clusters on each application. Similarly as for ten and six clusters, it consistently partitioned the data so that 13 clusters contained data for one or more species where in some cases, only a negligible amount of data was involved, e.g. 0.01%. The largest cluster contained all the remaining data, meaning data from nine or ten species were present in this very large cluster, between 33% and 50% of all the data were contained in this single cluster. More than 90% of the data representing single species was grouped into one cluster for 1 out of 10 species. With between 1 and 3 out of 10 species containing more than 80% of the corresponding data in a single cluster. (See Figure 4.16)

**Figure 4.14 ML Non-Overlapping Data - Six Clusters**

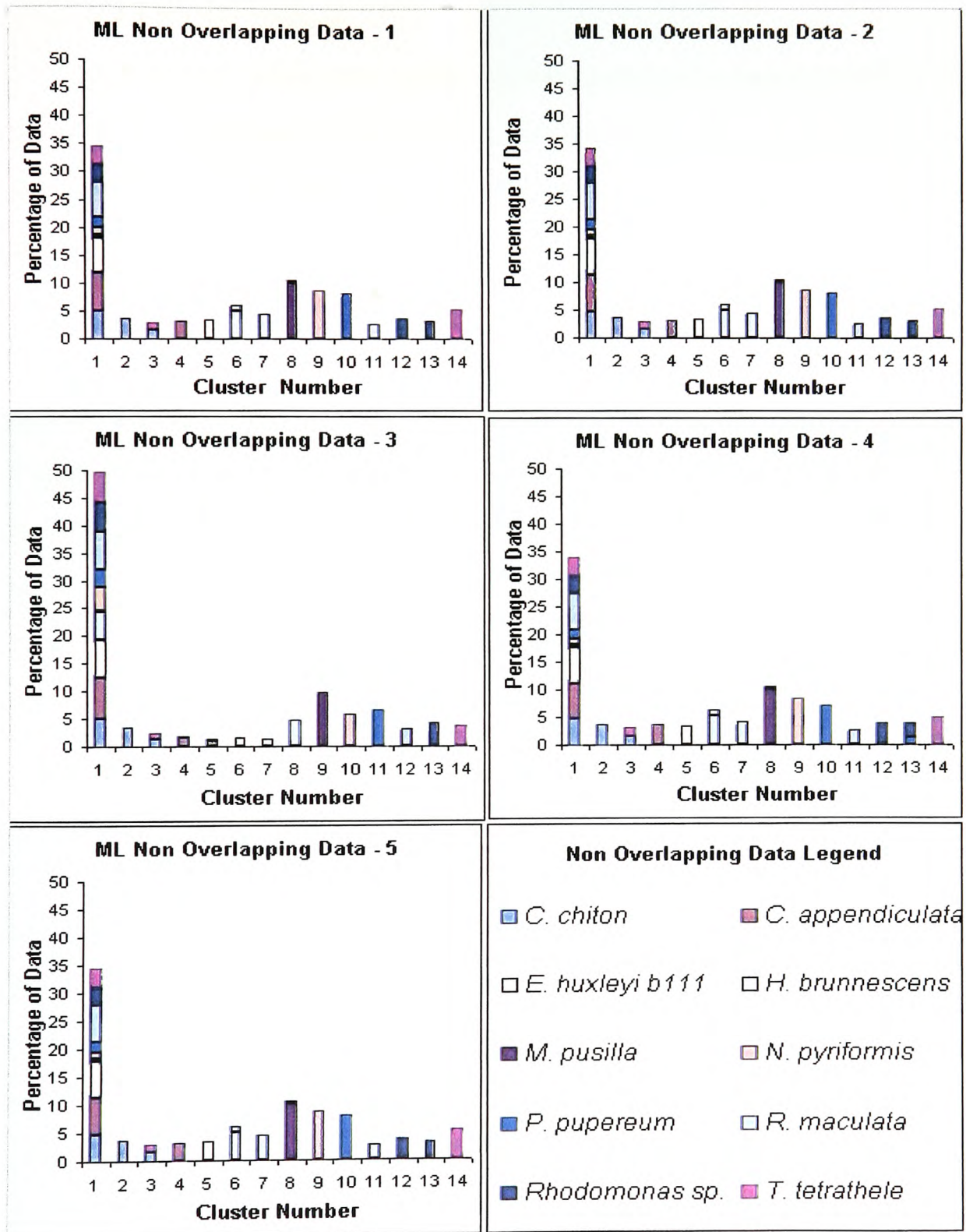


**Figure 4.15 ML Non-Overlapping Data – Ten Clusters**





**Figure 4.16 ML Non-Overlapping Data - Fourteen Clusters**



#### 4.4.1.2 Overlapping Data

##### 4.4.1.2.1 Six Clusters

In 3 out of 5 cases, one cluster was very large and contained data for all ten species, and between 50% and 54% of the data was contained in this cluster. More than 90% of the data representing single species was grouped into one cluster for between 0 and 9 out of 10 species. With between 2 and 9 out of 10 species containing more than 80% of the corresponding data in a single cluster. In 4 out of 5 cases, the data representing *G. veneficum*, *H. triquetra* and *A. carterae* are partitioned together, these species are in the same class. In 3 out of 5 applications, the data representing *A. coffaeformis* and *C. chiton* are grouped together, these species are in different classes but both belong to the group taxon. Also in 3 out of 5 cases, data representing *H. brunnescens* and *N. pyriformis* were grouped together, these species are in different classes and don't belong to the group taxon. (See Figure 4.17)

##### 4.4.1.2.2 Ten Clusters

One application of this algorithm resulted in nine clusters, not ten. The data for the ten species tended to be partitioned so that nine clusters contained some data from one of the nine, with the last cluster consistently containing data from all ten species in varying proportions in one very large cluster, just as with the non-overlapping data. The amount of data in the large cluster varies from 50% to 62%. More than 90% of the data representing single species was grouped into one cluster for between 1 and 3 out of 10 species. With between 2



and 5 out of 10 species containing more than 80% of the corresponding data in a single cluster. (See Figure 4.18)

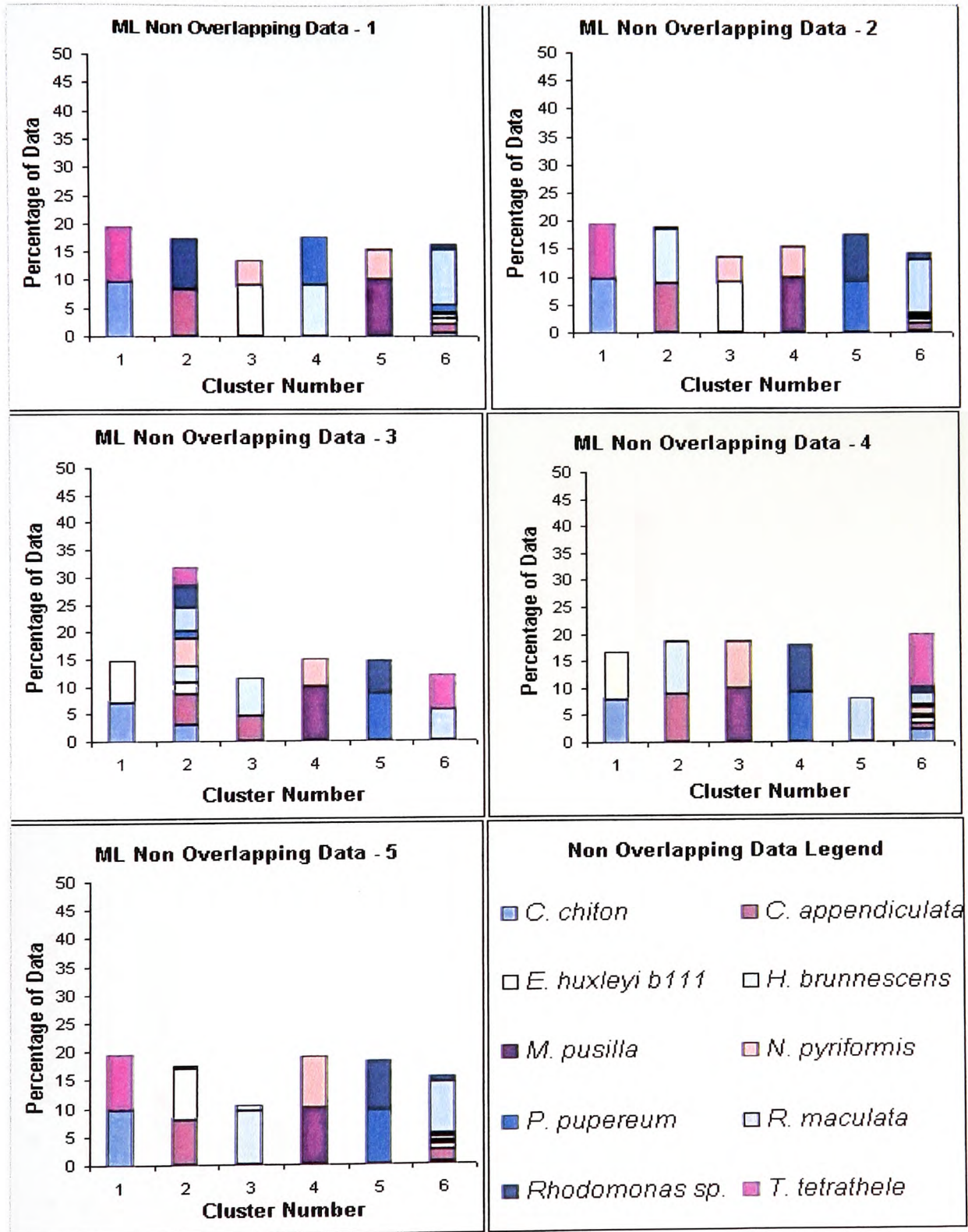
#### *4.4.1.2.3 Fourteen Clusters*

Data tended to only be grouped into eleven or twelve clusters instead of fourteen, with a further two or three clusters with very small amounts of data in them, therefore only eight or nine clusters were of any real interest. Again the last cluster contained data for all ten species in one large cluster. The amount of data in the large cluster varied between 58% and 60%. More than 90% of the data representing single species was grouped into one cluster for 2 out of 10 species. With 2 out of 10 species containing more than 80% of the corresponding data in a single cluster. (See Figure 4.19)

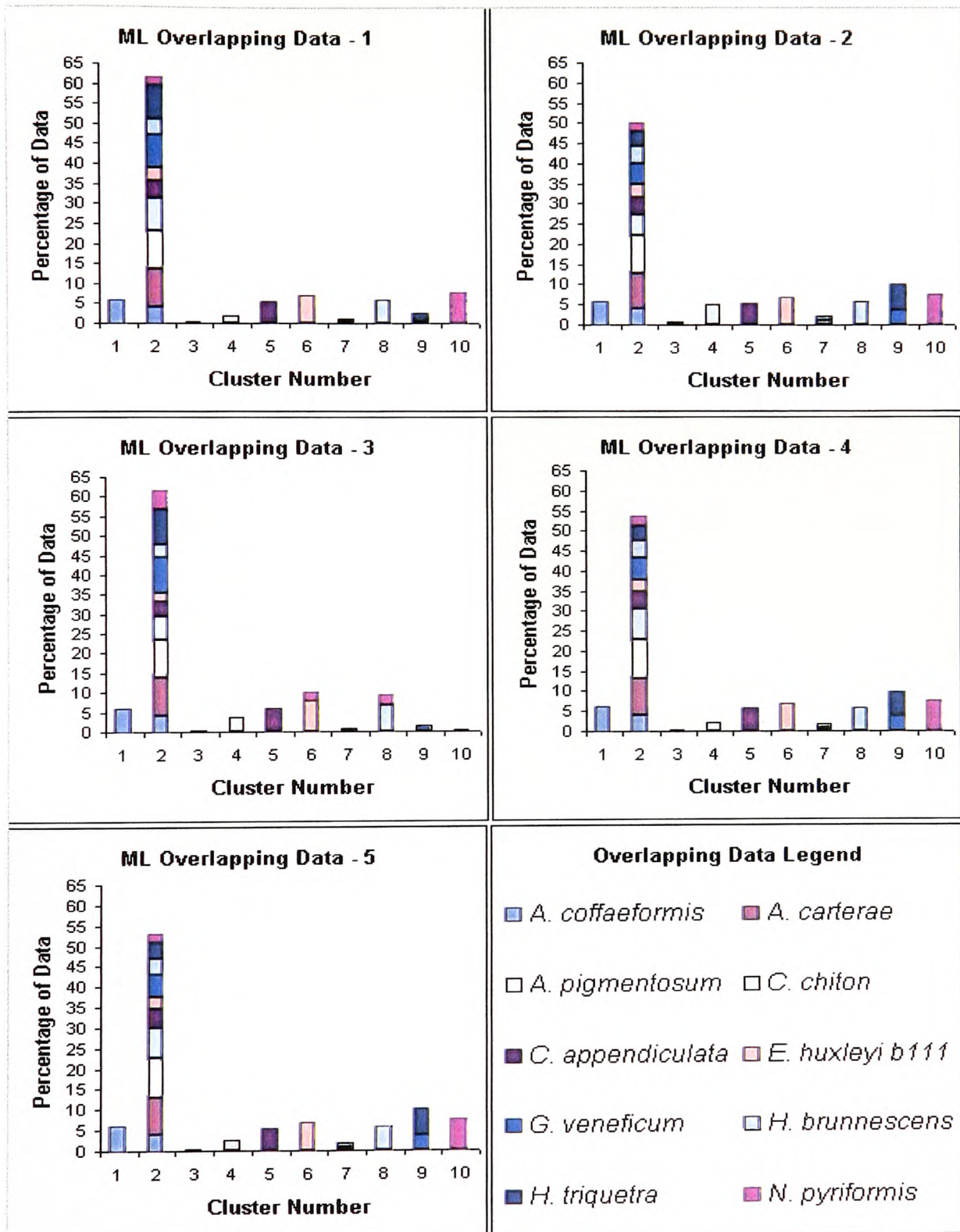
#### *4.4.1.2.4 Five Clusters – One Taxon*

The data in taxon one is partitioned into between 3 and 5 clusters in each application of the algorithm, between 90% and 100% of the group taxon data were partitioned in these clusters. In 4 out of 5 cases, data representing *A. pigmentosum* and *C. appendiculata* was grouped together, these species are in different classes. In 3 out of 5 applications, data representing *C. appendiculata* and *H. brunnescens*, which are also in different classes. (See Figure 4.20)

Figure 4.17 ML Overlapping Data - Six Clusters

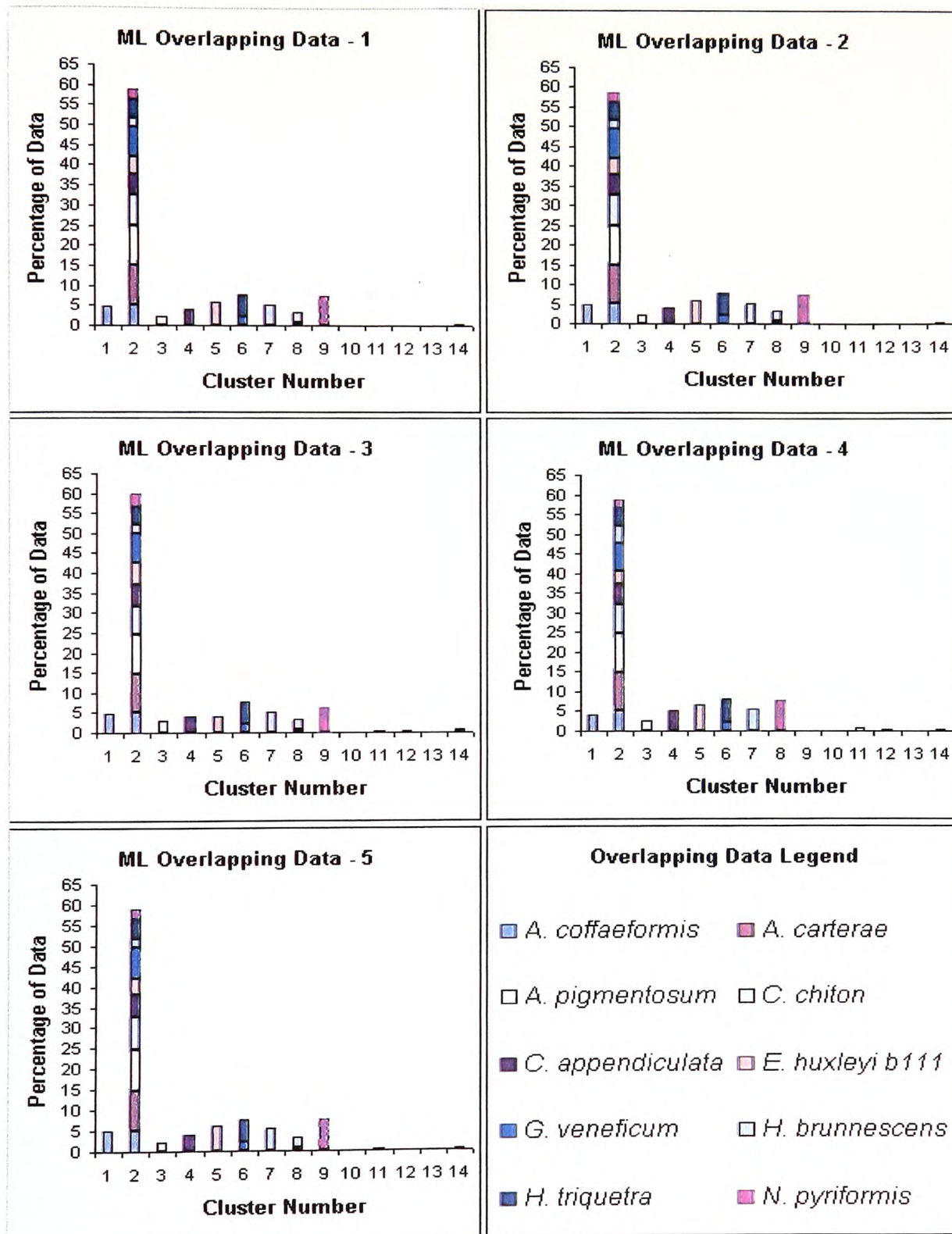


**Figure 4.18 ML Overlapping Data - Ten Clusters**

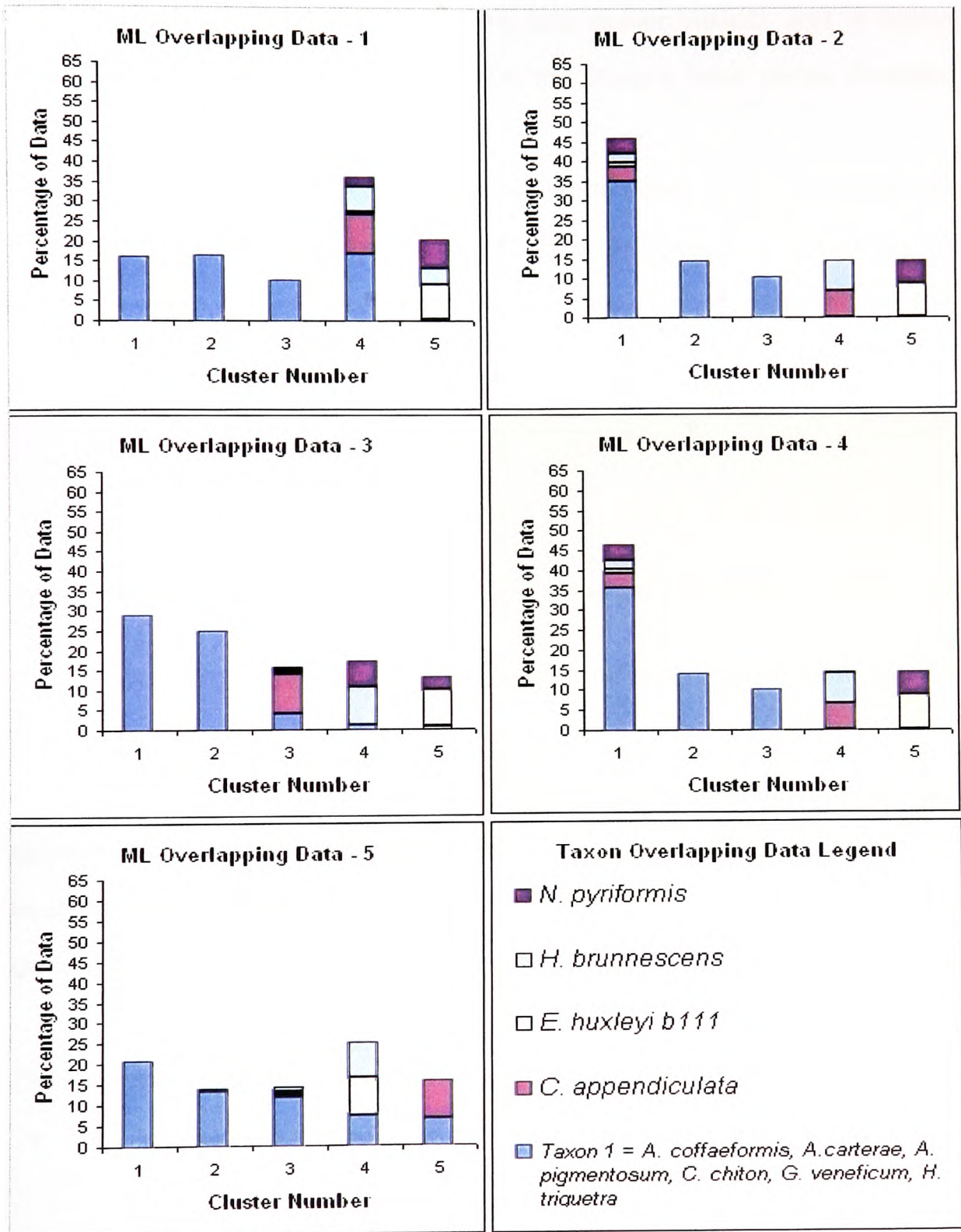




**Figure 4.19 ML Overlapping Data - Fourteen Clusters**



**Figure 4.20 ML Overlapping Data – Five Clusters / One Taxon**



#### 4.5 Minimum Total Volume MTV

The MTV algorithm minimises the total cluster volume, and is biased towards finding clustering solutions where the clusters have similar densities, rather than similar volumes.

This algorithm follows the notation in the paper by P. J. Rousseeuw published in 1996 and is obtained by changing the manner of calculating the quantities  $A_t$  and  $B_{it}$  which are variables used to represent the relevant objective function, so that [36]:

$$A_t = \frac{1}{2} \frac{\tau}{\beta} n_t^{p\beta-\tau-1} (\theta_t |\hat{S}_t|)^\beta,$$

$$B_{it} = n_t^{p\beta-e-1} (\theta_t |\hat{S}_t|)^\beta$$

where  $p$  is the data dimensionality,  $n_t = \sum_i u_{it}$ , and the parameters  $\beta$  and  $\tau$  are given by  $\beta = 1/2$ ,  $\tau = p/2$  where  $\beta$  and  $\tau$  are variables used in order that the formula can be generalised.

The focus of this method is to produce clustering solutions where the clusters have different cardinalities, and to minimise the total volume of the clusters. This is achieved by using equations  $A_t$  and  $B_{it}$  above which adapts the general algorithm to the MTV approach.

The Minimum Total Volume algorithm used is as follows:

1. For each cluster  $t$  initialise the cluster centre  $\mu_t$  to a randomly-selected data pattern and initialise the scatter matrix  $\hat{S}_t$  to the identity matrix

2. For each cluster  $t$ , calculate  $A_t$  and  $B_{it}$  which are variables used to simplify the objective function where,

$$A_t = \frac{1}{2} \frac{\tau}{\beta} n_t^{p\beta-\tau-1} (\theta_t |\hat{S}_t|)^\beta \text{ and}$$

$$B_{it} = n_t^{p\beta-\tau-1} (\theta_t |\hat{S}_t|)^\beta$$

3. Calculate memberships: for each data pattern  $x_i$ :

- (i) Initialise  $T_i$  to an empty set where  $T_i$  is a set of indices  $t$  of the clusters that data pattern  $x_i$  belongs to,
- (ii) For each cluster  $t$  calculate the distance from each data pattern  $x_i$  to the cluster centre  $B_{it} \leftarrow B_t d_{it}^2$
- (iii) For each cluster  $t$  calculate the membership  $u_{it}$  of each data pattern  $x_i$  to the cluster where,

$$u_{it} \leftarrow \frac{1/B_{it}}{\sum_{r \in T_i} 1/B_{ir}} - \frac{1}{B_{it}} \left[ \frac{\sum_{r \in T_i} A_r / B_{ir}}{\sum_{r \in T_i} 1/B_{ir}} - A_t \right], t \in T_i$$

$$u_{it} \leftarrow 0, t \notin T_i$$

- (iv) If any  $u_{it} > 0$ , add  $t$  to set  $T_i$ , and repeat step 3 while some memberships are strictly negative.

4. For each cluster  $t$  update the cluster centre  $\mu_t$  and the scatter matrix

$\hat{S}_t$  as follows,

$$\mu_i \leftarrow \frac{\sum_i u_{ii}^2 \mathbf{x}_i}{\sum_i u_{ii}^2} \quad \hat{\mathbf{S}}_i \leftarrow \frac{\sum_i u_{ii}^2 (\mathbf{x}_i - \mu_i)(\mathbf{x}_i - \mu_i)^T}{\sum_i u_{ii}^2}$$

5. If no membership  $u_{ii}$  changed by more than a small value  $\varepsilon$  during the previous iteration, stop; otherwise return to step 2.

Here, a reasonable *a priori* assumption is that all clusters should start with the same volume; i.e.  $\theta_i = 1$  for all  $t$  and the value of  $\varepsilon$  indicating convergence was taken to be 0.01.



## **4.5.1 Results**

### *4.5.1.1 Non-Overlapping Data*

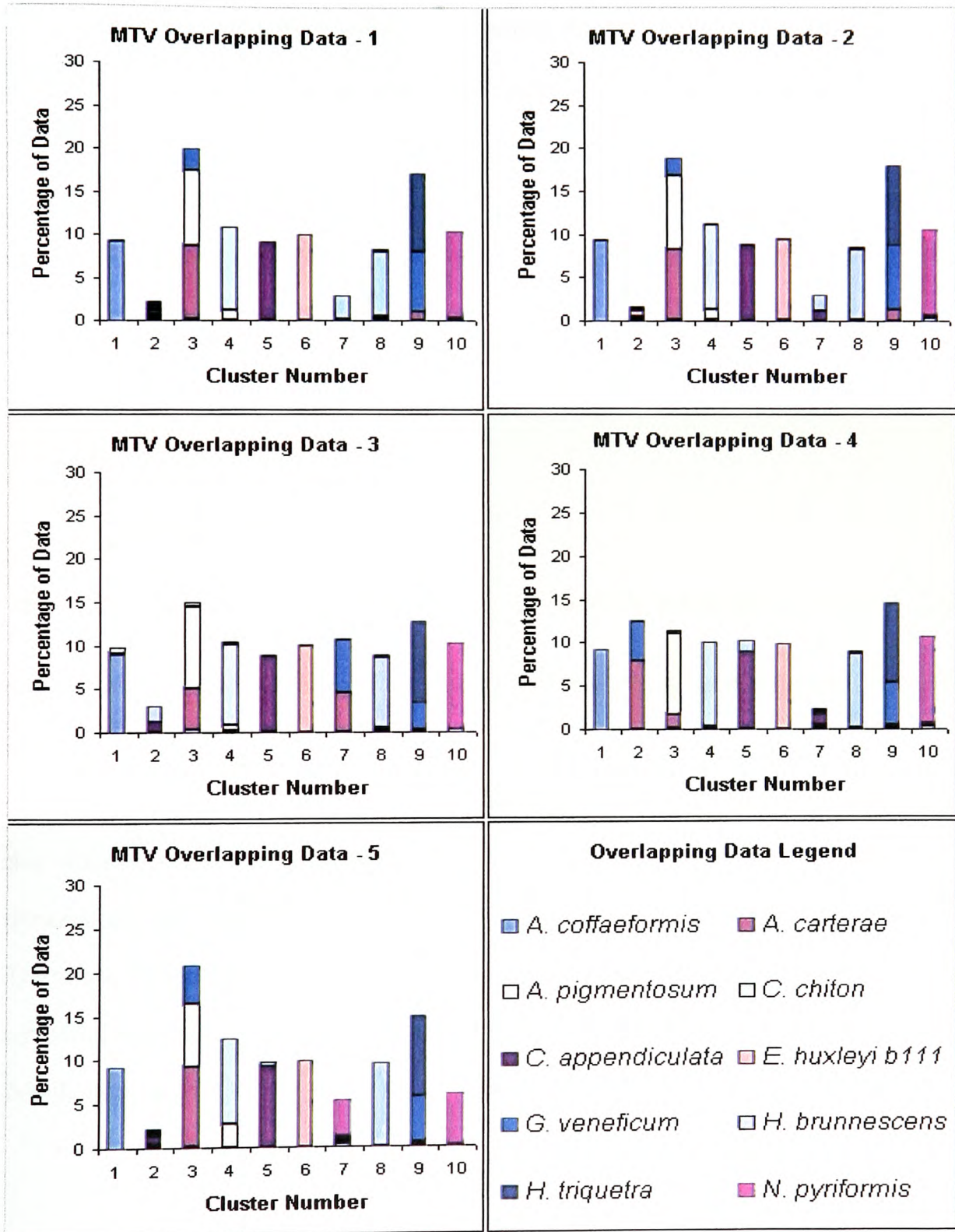
No results were produced by this algorithm using the Non-Overlapping Data set. MTV failed to converge to a solution, both when the number of clusters specified was the actual number of clusters present, and when it differed from the optimal or 'natural' number of clusters. Further investigation showed that this algorithm tended to become trapped cycling between two clustering solutions.

### *4.5.1.2 Overlapping Data*

#### *4.5.1.2.1 Ten Clusters*

Only results where ten clusters were specified converged. There are no results for the cases where fourteen clusters, six clusters or the group taxon specified, when using the overlapping data set with this algorithm. The data was partitioned into ten clusters each time. More than 90% of the data representing single species was grouped into one cluster for between 5 and 7 out of 10 species. With between 7 and 9 out of 10 species containing more than 80% of the corresponding data in a single cluster. In all 5 cases, data representing *A. carterae* and *A. pigmentosum*, *G. veneficum* and *H. triquetra*, and *A. carterae* and *G. veneficum* were grouped together in the relevant pairings, pair-wise these species are from the same class. In 4 out of 5 cases, data representing *C. appendiculata* and *H. brunnescens* were partitioned together, these species are also in the same class. (See Figure 4.21)

**Figure 4.21** MTV Overlapping Data – Ten Clusters



#### 4.6 Sum of All Normalised Determinants SAND

The SAND algorithm attempts to reduce the bias of the MTV algorithm towards finding clustering solutions where the clusters have similar densities, by normalising the cluster volumes by the dimensionality.

This algorithm follows the notation in the paper by P. J. Rousseeuw published in 1996 and is obtained by changing the manner of calculating the quantities  $A_t$  and  $B_{it}$  which are variables used to represent the relevant objective function, so that [36]:

$$A_t = \frac{1}{2} \frac{\tau}{\beta} n_t^{p\beta-\tau-1} (\theta_t | \hat{\mathbf{S}}_t |)^{\beta},$$

$$B_{it} = n_t^{p\beta-e-1} (\theta_t | \hat{\mathbf{S}}_t |)^{\beta}$$

where  $p$  is the data dimensionality,  $n_t = \sum_i u_{it}$ , and the parameters  $\beta$  and  $\tau$  are given by  $\beta = 1/p$ ,  $\tau = 1$  where  $\beta$  and  $\tau$  are variables used in order that the formula can be generalised.

The focus of this method is similar to that of MTV, but uses the  $p$ th root of the volume, not the squared root. This amount is very small with seven dimensions, and it gives an average size of the clusters instead of the volume. Therefore SAND tends to minimise the total size of the clusters. This is achieved by using equations  $A_t$  and  $B_{it}$  above which adapts the general algorithm to the SAND approach.

The Sum of All Normal Determinants algorithm used is as follows:

1. For each cluster  $t$  initialise the cluster centre  $\mu_t$  to a randomly-selected data pattern and initialise the scatter matrix  $\hat{S}_t$  to the identity matrix
2. For each cluster  $t$ , calculate  $A_t$  and  $B_{it}$  which are variables used to simplify the objective function where,

$$A_t = \frac{1}{2} \frac{\tau}{\beta} n_t^{p\beta-\tau-1} (\theta_t | \hat{S}_t |)^{\beta} \text{ and}$$

$$B_{it} = n_t^{p\beta-e-1} (\theta_t | \hat{S}_t |)^{\beta}$$

3. Calculate memberships: for each data pattern  $x_i$ :

- (i) Initialise  $T_i$  to an empty set where  $T_i$  is a set of indices  $t$  of the clusters that data pattern  $x_i$  belongs to,
- (ii) For each cluster  $t$  calculate the distance from each data pattern  $x_i$  to the cluster centre  $B_{it} \leftarrow B_t d_{it}^2$
- (iii) For each cluster  $t$  calculate the membership  $u_{it}$  of each data pattern  $x_i$  to the cluster where,

$$u_{it} \leftarrow \frac{1/B_{it}}{\sum_{r \in T_i} 1/B_{ir}} - \frac{1}{B_{it}} \left[ \frac{\sum_{r \in T_i} A_r / B_{ir}}{\sum_{r \in T_i} 1/B_{ir}} - A_t \right], t \in T_i$$

$$u_{it} \leftarrow 0, t \notin T_i$$

- (iv) If any  $u_{it} > 0$ , add  $t$  to set  $T_i$ , and repeat step 3 while some memberships are strictly negative.

4. For each cluster  $t$  update the cluster centre  $\mu_t$  and the scatter matrix  $\hat{S}_t$  as follows,

$$\mu_t \leftarrow \frac{\sum_i u_{it}^2 \mathbf{x}_i}{\sum_i u_{it}^2} \quad \hat{S}_t \leftarrow \frac{\sum_i u_{it}^2 (\mathbf{x}_i - \mu_t)(\mathbf{x}_i - \mu_t)^T}{\sum_i u_{it}}$$

5. If no membership  $u_{it}$  changed by more than a small value  $\varepsilon$  during the previous iteration, stop; otherwise return to step 2.

Here, a reasonable *a priori* assumption is that all clusters should start with the same volume; i.e.  $\theta_t = 1$  for all  $t$ , and the value of  $\varepsilon$  indicating convergence was taken to be 0.01.

## **4.6.1 Results**

### *4.6.1.1 Non-Overlapping Data*

#### *4.6.1.1.1 Six Clusters*

This algorithm partitioned the data in six clusters on each application. More than 90% of the data representing single species was grouped into one cluster for between 7 and 9 out of 10 species. With between 9 and 10 out of 10 species containing more than 80% of the corresponding data in a single cluster. In 4 out of 5 cases, data representing *M. pusilla*, and *N. pyriformis* were grouped together, these species belong to the same class. Also in 4 out of 5 applications, data representing *E. huxleyi* and *N. pyriformis* were partitioned in to the same cluster, these species are in different classes. In 3 out of the 5 cases, data representing *C. appendiculata* and *Rhodomonas sp.* were grouped together, these species also belong to different classes. There was a variety in the pattern of the groupings over the five applications. (See Figure 4.22)

#### *4.6.1.1.2 Ten Clusters*

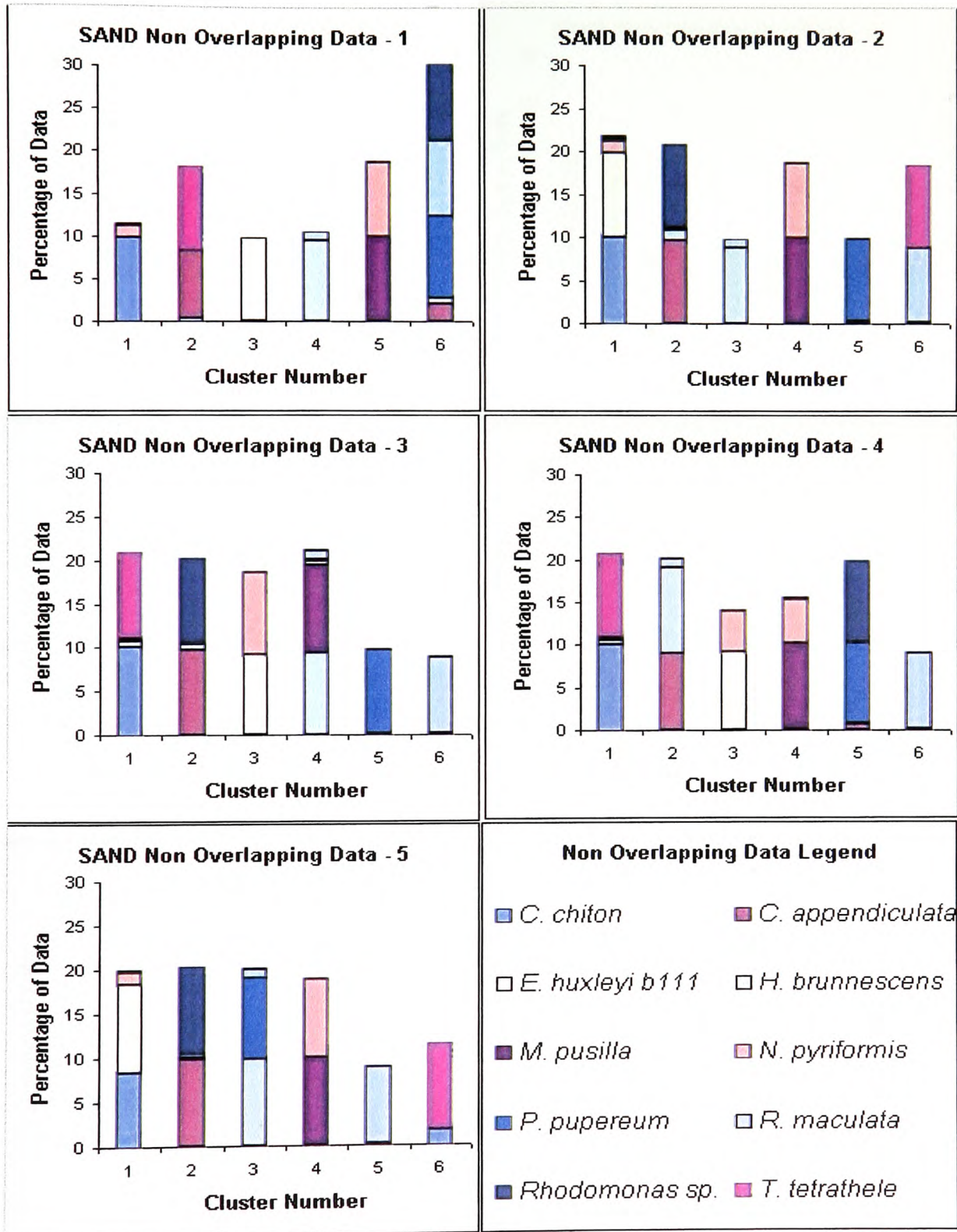
This algorithm partitioned the data in ten clusters on each application, although one cluster in the first application only contained 0.08% of the data. More than 90% of the data representing single species was grouped into one cluster for between 6 and 7 out of 10 species. With between 8 and 10 out of 10 species containing more than 80% of the corresponding data in a single cluster. In all 5 cases, data representing *H. brunnescens* and *R. maculata*, were grouped together, these species are from different classes. Also in all 5 cases, data

representing *M. pusilla*, and *N. pyriformis* and *C. appendiculata*, *H. brunnescens* and *Rhodomonas sp.* were grouped together, the species in each combination belong to the same class. In 3 out of the 5 cases, data representing *C. chiton* and *T. tetrathele* were grouped together, these species belong to different classes. (See Figure 4.23)

#### 4.6.1.1.3 Fourteen Clusters

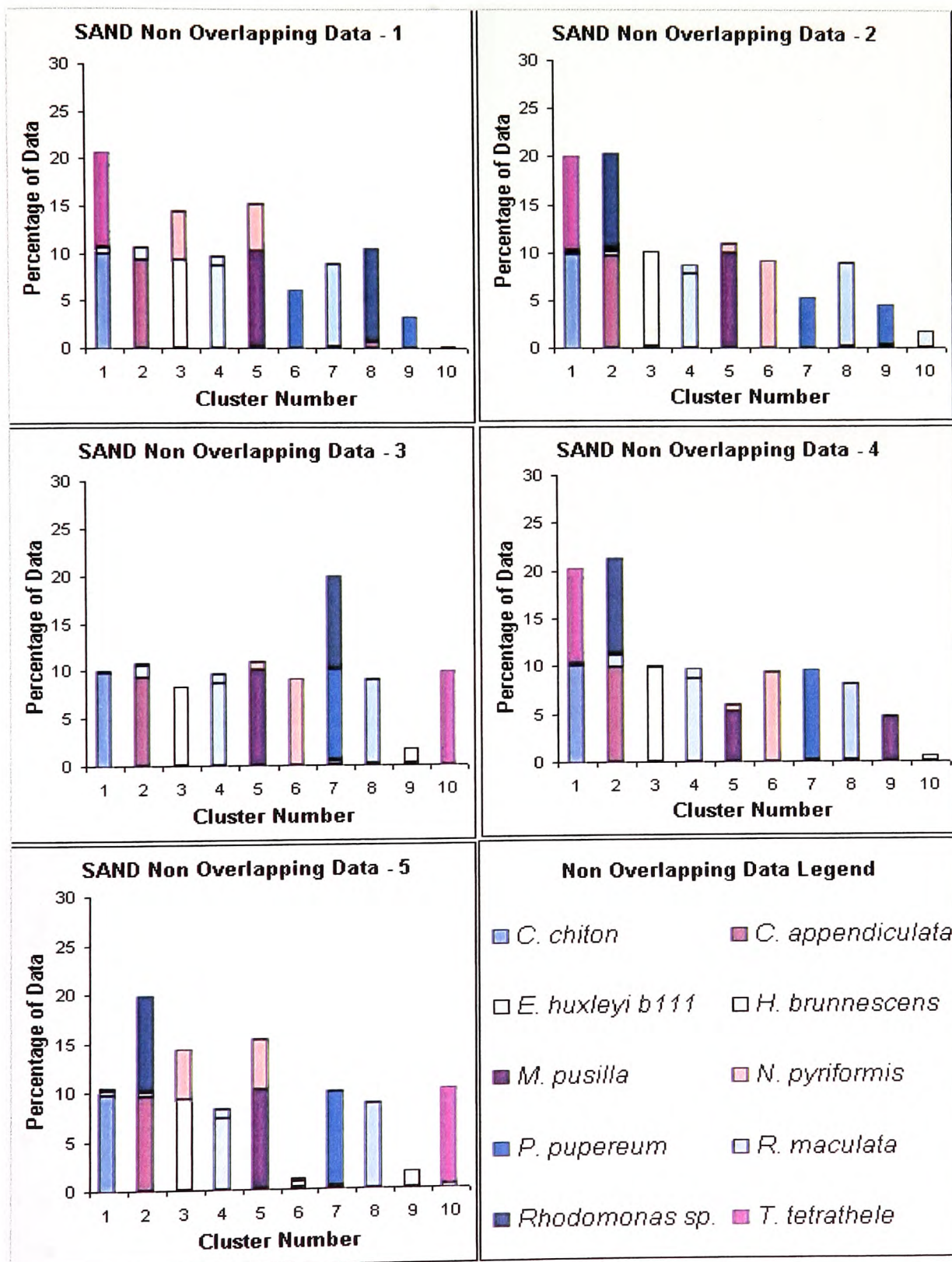
This algorithm partitioned the data in fourteen clusters on 4 occasions with one cluster in one case only containing 0.02% of the data. Only 12 clusters were created in the 5<sup>th</sup> application. More than 90% of the data representing single species was grouped into one cluster for between 3 and 7 out of 10 species. With between 5 and 9 out of 10 species containing more than 80% of the corresponding data in a single cluster. In all 5 cases, data representing *H. brunnescens* and *R. maculata*, were grouped together, these species are from different classes. In 3 out of 5 cases, data representing *M. pusilla*, and *N. pyriformis* were grouped together, these species belong to the same class. Also in 3 out of the 5 cases, data representing *C. appendiculata* and *H. brunnescens* were grouped together, these species also belong to the same class. (See Figure 4.24)

**Figure 4.22 – SAND Non-Overlapping Data - Six Clusters**

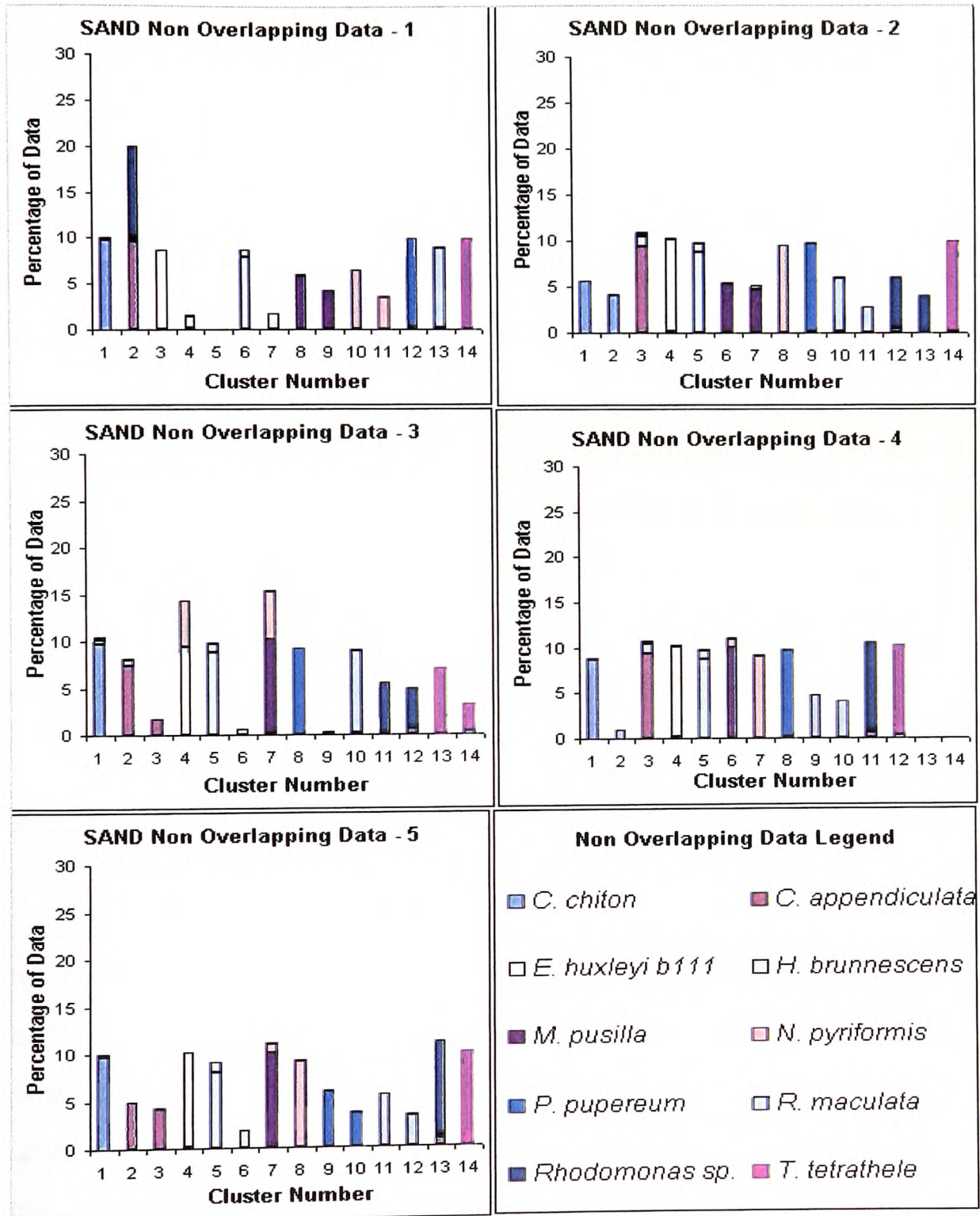




**Figure 4.23 SAND Non-Overlapping Data - Ten Clusters**



**Figure 4.24 SAND Non-Overlapping Data - Fourteen Clusters**



#### 4.6.1.2 Overlapping Data

##### 4.6.1.2.1 Six Clusters

All five applications of this algorithm partitioned the data into 6 clusters. More than 90% of the data representing single species was grouped into one cluster for between 6 and 8 out of 10 species. With between 8 and 9 out of 10 species containing more than 80% of the corresponding data in a single cluster. In all 5 cases, data representing *C. appendiculata* and *G. veneficum* were partitioned together, both of these species are in the same class. Also looking at all 5 cases, various combinations of *A. coffaeformis*, *A. carterae*, *A. pigmentosum*, *C. chiton*, *G. veneficum* and *H. triquetra* were grouped together in each case, these species are not all from the same class but all belong to the group taxon. In 3 out of 5 applications, data representing *E. huxleyi* and *N. pyriformis* were grouped together in the same cluster, these species are in different classes. (See Figure 4.25)

##### 4.6.1.2.2 Ten Clusters

One of the five applications only partitioned the data into nine clusters, not ten, but the pattern of the data clustering was similar in each case. The data was partitioned into ten clusters each time. More than 90% of the data representing single species was grouped into one cluster for between 4 and 6 out of 10 species. With between 6 and 9 out of 10 species containing more than 80% of the corresponding data in a single cluster. In all 5 cases, data representing *A. coffaeformis*, *A. carterae*, *A. pigmentosum*, and *G. veneficum* were partitioned

together, not all of these species are in the same class, but they are all in the group taxon. Also in all 5 cases, *H. triquetra* and *G. veneficum* were grouped together, these species are from the same class. In 3 out of 5 cases, data representing *C. appendiculata* and *H. brunnescens* were partitioned together, these species are also in the same class. (See Figure 4.26)

#### 4.6.1.2.3 Fourteen Clusters

Two out of the five applications only had thirteen clusters created instead of fourteen, and one of the cases had fourteen clusters, but one of the clusters only contained 0.02% of the data and therefore appeared to be empty. More than 90% of the data representing single species was grouped into one cluster for between 4 and 6 out of 10 species. With between 7 and 8 out of 10 species containing more than 80% of the corresponding data in a single cluster. In all 5 cases, data representing *A. carterae*, *A. pigmentosum*, and *G. veneficum* were partitioned together, all of these species are in the same class. Also in all 5 cases, *H. triquetra* and *G. veneficum* were grouped together, these species are also from the same class. (See Figure 4.27)

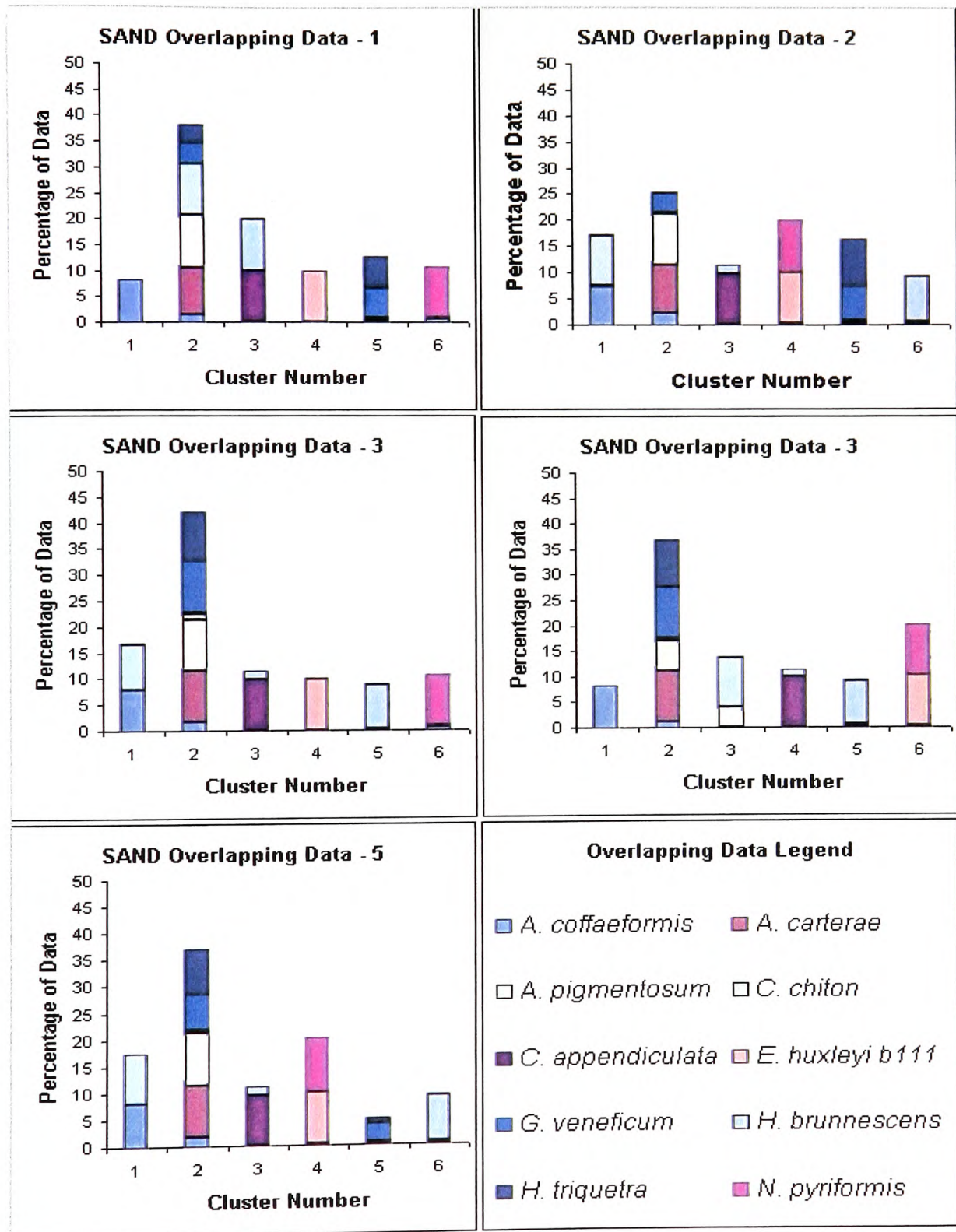
#### 4.6.1.2.4 Five Clusters - One Taxon

At least 98% of the taxon data was partitioned into 2 - 3 main clusters out of the five. In 4 out of 5 results, data representing *C. appendiculata* and *H. brunnescens* were partitioned in the same cluster, these species are in the same class. In 3 out of 5 results, data representing *E. huxleyi* and *N. pyriformis* were

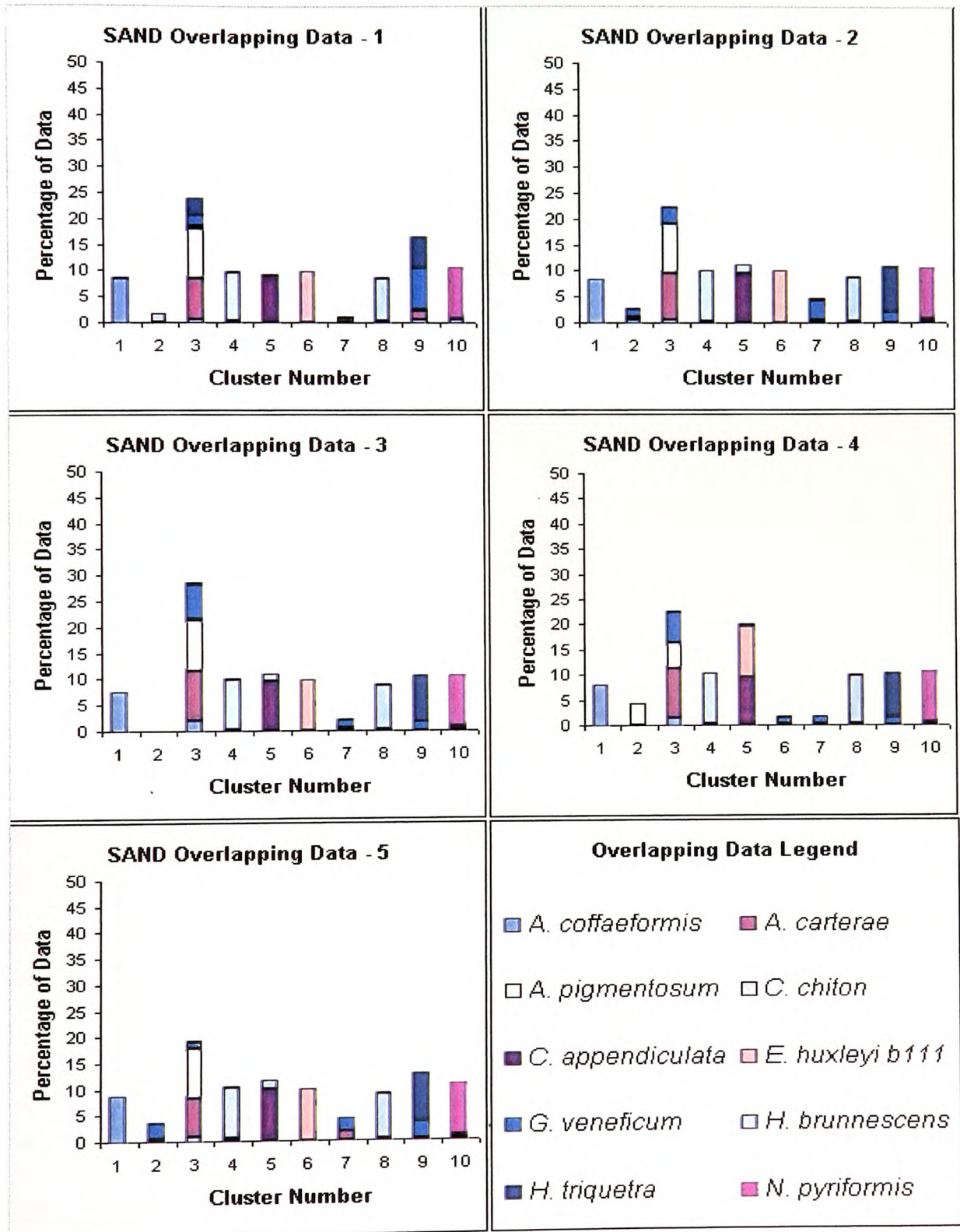
partitioned in the same cluster, these species are in different classes. (See Figure 4.28)



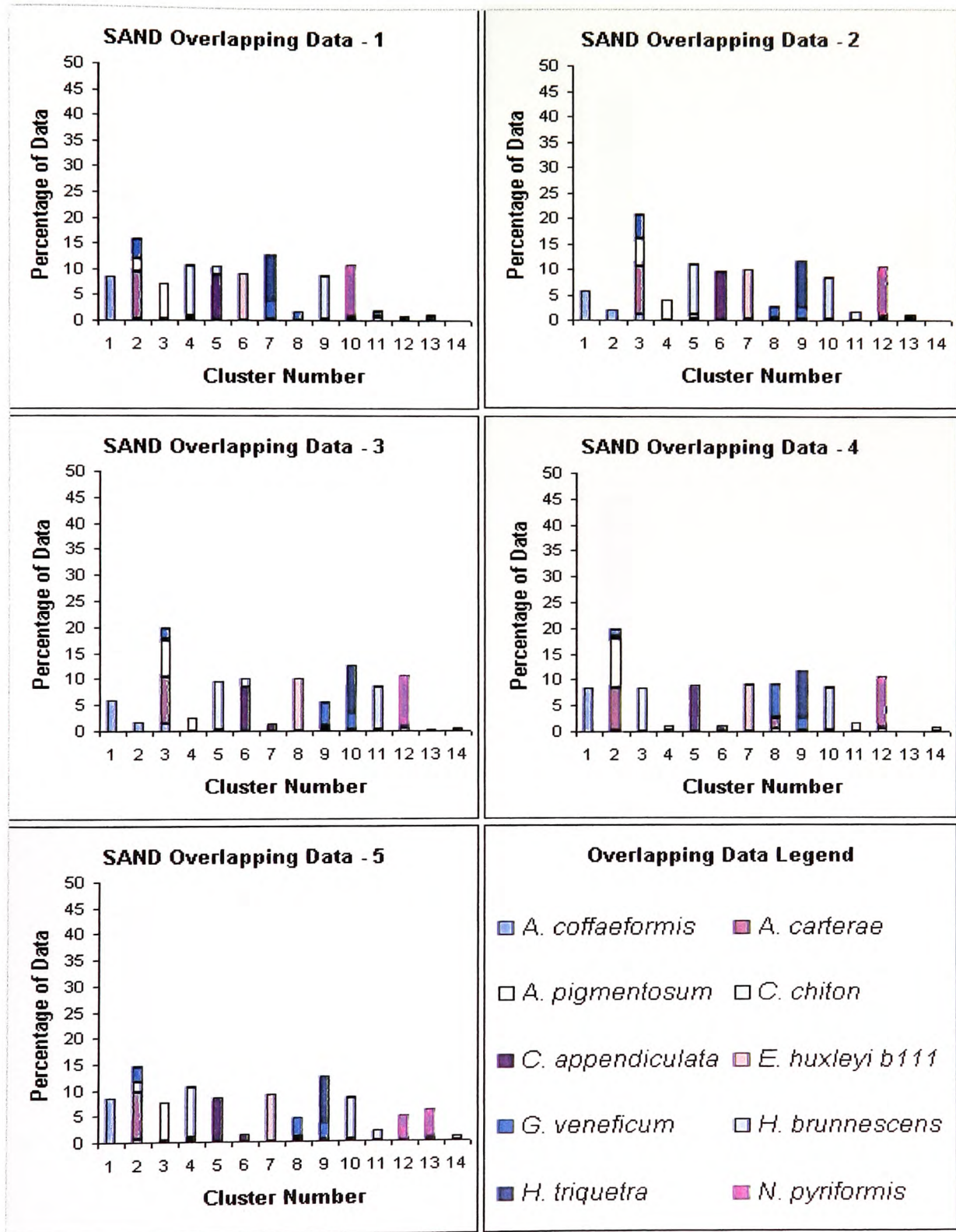
**Figure 4.25 SAND Overlapping Data - Six Clusters**



**Figure 4.26 SAND Overlapping Data - Ten Clusters**

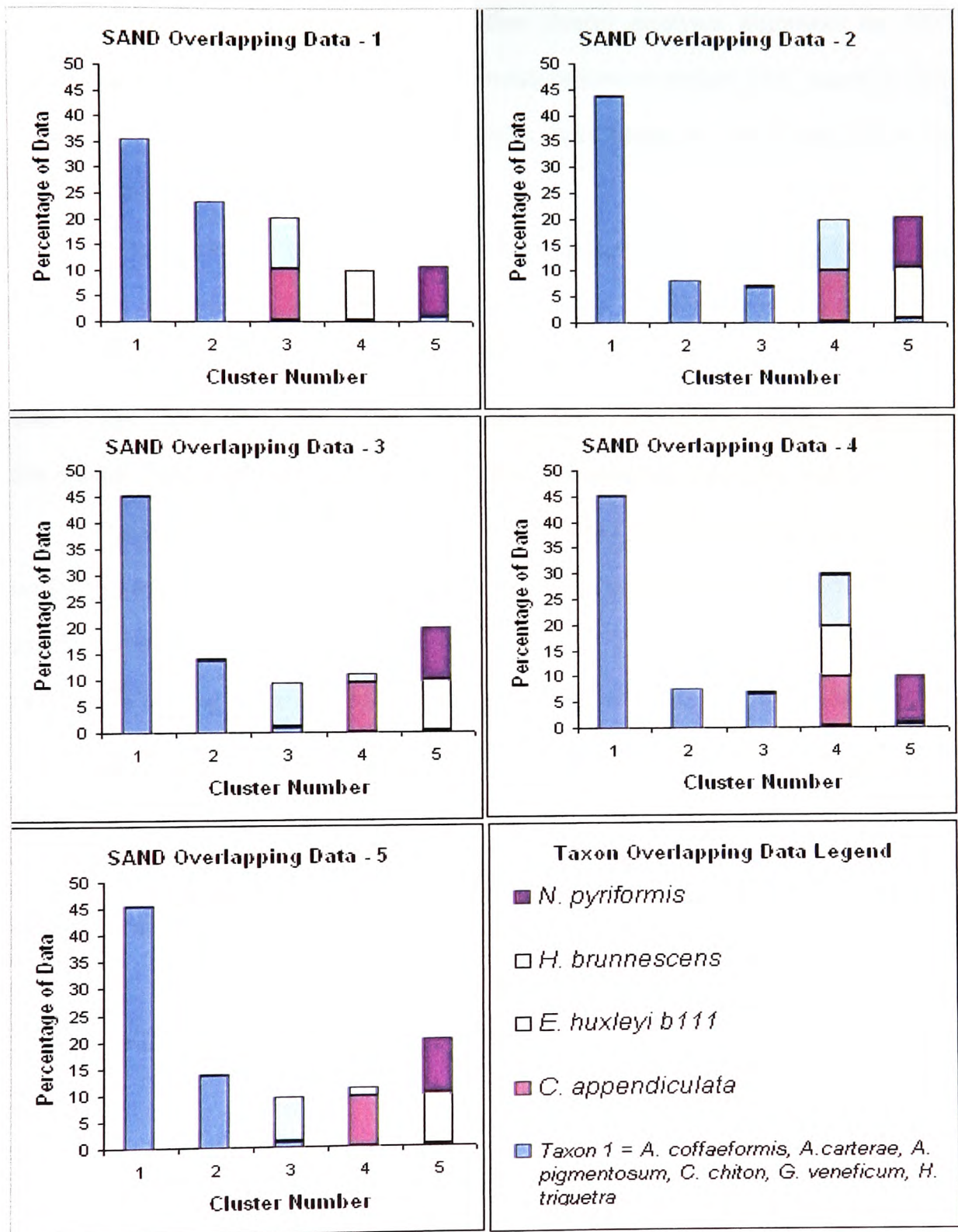


**Figure 4.27 SAND Overlapping Data - Fourteen Clusters**





**Figure 4.28 SAND Overlapping Data – Five Clusters / One Taxon**



#### **4.7 DKLL (Demers Kim Legendre Legendre - Critical Distances) Algorithm**

Demers *et al.* proposed an iterative cluster analysis algorithm for AFC data capable of modelling the heterogeneous nature of typical AFC clusters [35]. This algorithm, henceforth denoted as DKLL, was based on an extension of the classic *k*-means algorithm [51].

“A common problem with such iterative algorithms is that over the course of several iterations, one cluster tends to expand to include all the data. The DKLL algorithm incorporated the use of threshold “critical distances” to prevent this. At each step of the algorithm, only those points within the critical distance of the cluster centre are used in calculating the corresponding scatter matrix.

The critical distances are gradually increased during successive iterations, under the assumption that the clustering becomes progressively more reliable. In practice, however, this approach is not ideal: there are no guidelines as to how rapidly the critical distances should be increased, and the number of data points within the critical distance of the cluster centre may well fall to zero at some point, leaving the cluster scatter matrix undefined.

Furthermore, the algorithm frequently fails to converge to an optimal clustering solution, requiring user intervention to reject inadequate results [35].

The DKLL algorithm described in Demers *et al.* can be summarised as follows [35]:

1. For each cluster  $t$  initialise its cluster centre  $\mu_t$  to a randomly-selected data pattern and initialise its scatter matrix  $\hat{S}_t$  to the identity matrix

2. Assign each data pattern  $\mathbf{x}_i$  to its closest cluster  $t$ ; i.e. for which  $d_{it} = \min_k d_{ik}$
3. For each cluster  $t$ , update  $\mu_t$  to the mean of the set of patterns assigned to cluster  $t$
4. For each cluster  $t$ , calculate the 'critical distance' (Euclidean distance to the closest neighbouring cluster centre)  $c_t = \min_{k \neq t} \|\mu_k - \mu_t\|$
5. For each cluster  $t$ , update  $\hat{S}_t$  to the covariance matrix of the set of patterns assigned to cluster  $t$  and falling within a Euclidean distance  $c_{t,z}$  of the cluster centre.
6. Calculate the objective criterion  $F = \sum_i d_i^2$  where  $d_i$  is the generalised distance between pattern  $i$  and the centroid of the cluster to which it has been assigned.
7. Repeat steps 2-6 until  $F$  attains a minimum value.

The quantity  $z$  is initially small (around 0.25); this is designed to exclude patterns for which the cluster identity is uncertain from influencing the calculation of the scatter matrices. As the algorithm progresses, the cluster assignments are supposed to become more reliable and  $z$  is increased accordingly. In the implementation used here,  $z$  is increased by 0.005 at each iteration, up to a maximum value of 0.5. Any cluster with less than  $3p$  data patterns assigned to it following step 2 (where  $p$  is the data dimensionality) is deemed to be irrelevant and is removed from further consideration" [2].

## **4.7.1 Results**

### **4.7.1.1 Non-Overlapping Data**

#### **4.7.1.1.1 Six Clusters**

More than 90% of the data representing single species was grouped into one cluster for 9 out of 10 species. With 10 out of 10 species containing more than 80% of the corresponding data in a single cluster. In 4 out of 5 applications, data representing *H. brunnescens* and *R. maculata* were grouped together, these species are in different classes. Also in 4 out of 5 cases, data representing *C. chiton* and *T. tetrathele* were partitioned into the same cluster, these species are also in different classes. Again, in 4 out of 5 applications, data representing *C. appendiculata*, *P. pupureum*, and *Rhodomonas sp.* were grouped together, and again these species are in different classes. In 3 out of 5 applications, data representing *M. pusilla* and *N. pyriformis* were partitioned in the same cluster, these species are in the same class. (See Figure 4.29)

#### **4.7.1.1.2 Ten Clusters**

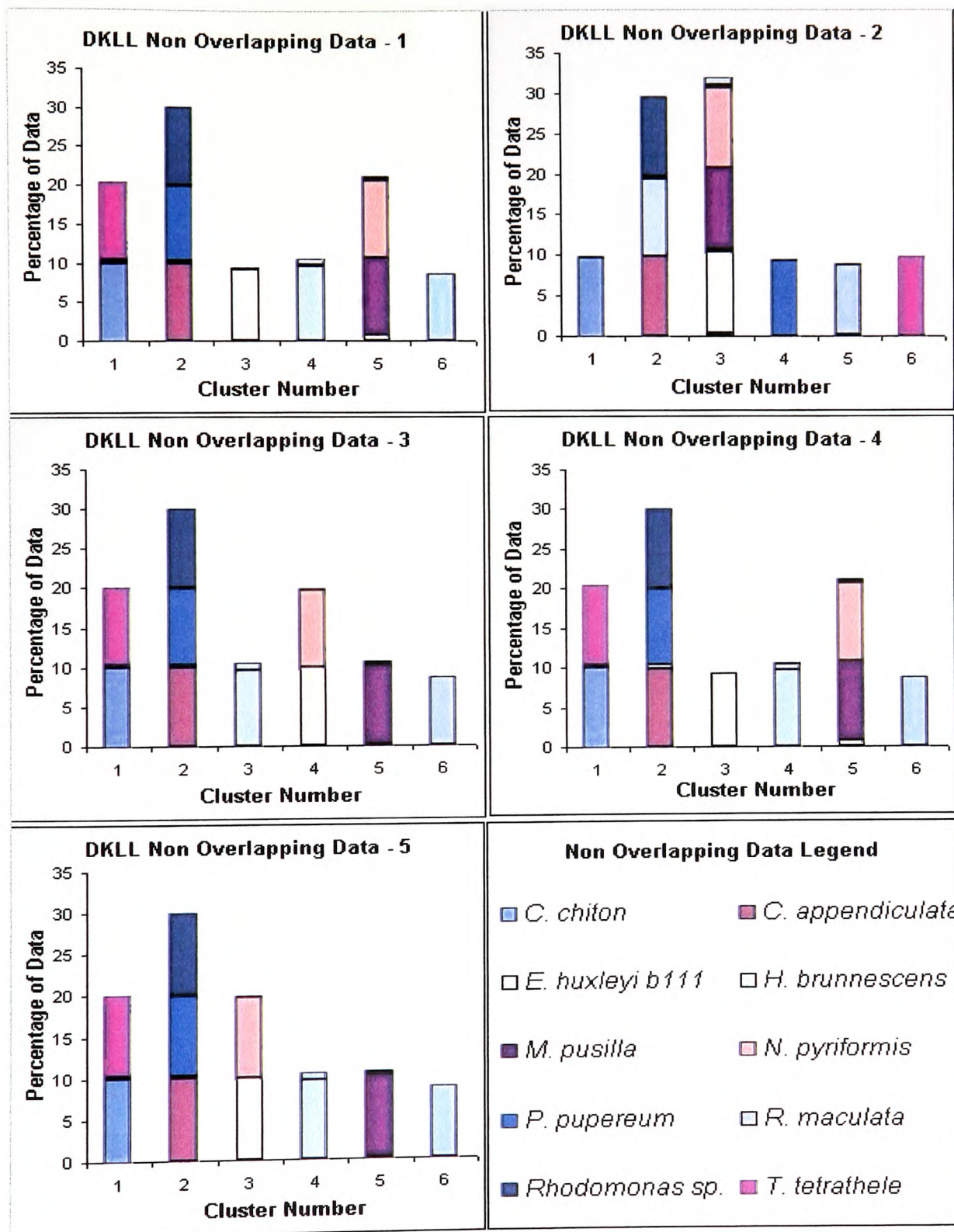
Ten clusters were not always generated, and in fact this algorithm was applied many times to get five sets of results with ten clusters. More than 90% of the data representing single species was grouped into one cluster for between 7 and 9 out of 10 species. With between 8 and 9 out of 10 species containing more than 80% of the corresponding data in a single cluster. In 3 out of 5 applications, data representing *C. chiton* and *T. tetrathele* were grouped together as was data

representing *H. brunnescens* and *R. maculata*, both pairs of species were in different classes. (See Figure 4.30)

#### 4.7.1.1.3 Fourteen Clusters

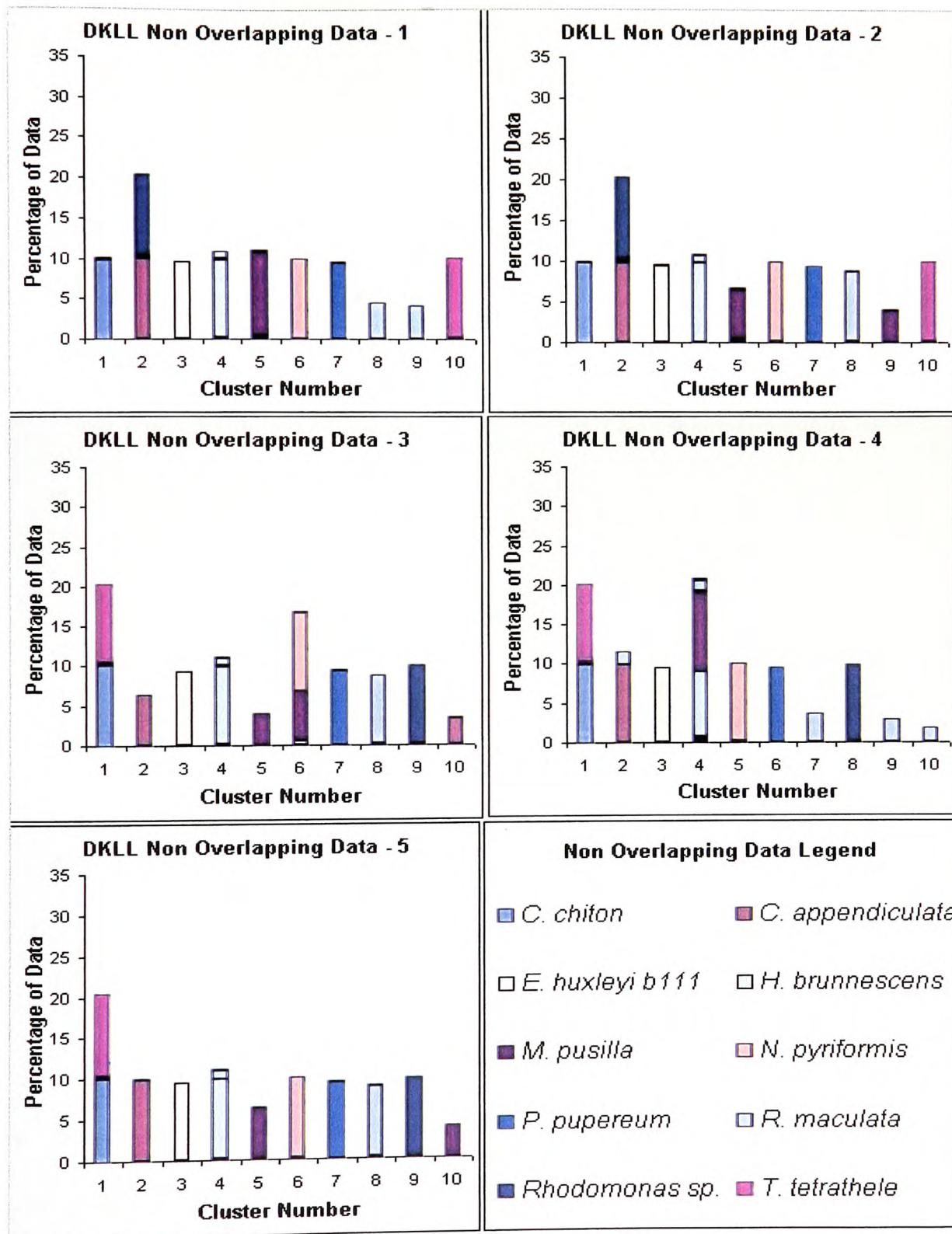
Between eleven and fourteen clusters were generated over the 5 applications of this algorithm. More than 90% of the data representing single species was grouped into one cluster for between 5 and 8 out of 10 species. With between 6 and 9 out of 10 species containing more than 80% of the corresponding data in a single cluster. In 4 out of 5 applications, data representing *H. brunnescens* and *R. maculata* were grouped together, these species were in different classes. (See Figure 4.31)

**Figure 4.29 DKLL Non-Overlapping Data - Six Clusters**

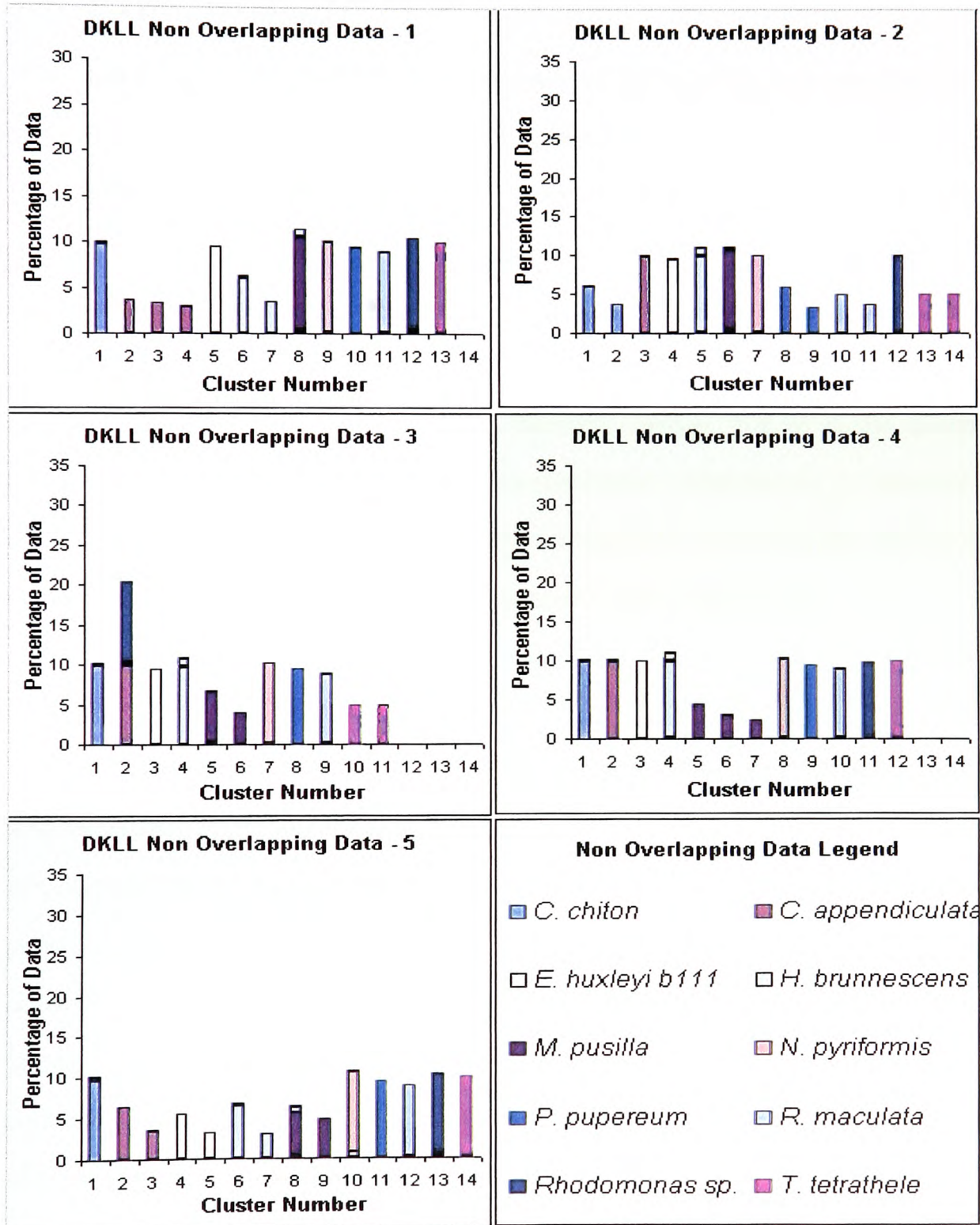




**Figure 4.30 DKLL Non-Overlapping Data - Ten Clusters**



**Figure 4.31 DKLL Non-Overlapping Data - Fourteen Clusters**





#### 4.7.1.2 Overlapping Data

##### 4.7.1.2.1 Six Clusters

More than 90% of the data representing single species was grouped into one cluster for between 9 and 10 out of 10 species. With between 9 and 10 out of 10 species containing more than 80% of the corresponding data in a single cluster. In all 5 applications, data representing *A. carterae*, *G. veneficum*, and *H. triquetra* were grouped together, all three species are from the same class. Again in all 5 cases, data representing *A. carterae*, *A. pigmentosum* and *C. chiton* were grouped together, these species are in different classes, but all in the group taxon. In 3 out of 5 applications, data representing *A. coffaeformis*, *A. carterae*, *G. veneficum*, and *H. triquetra* were grouped together, these species are from different classes but all belong to the group taxon. (See Figure 4.32)

##### 4.7.1.2.2 Ten Clusters

The pattern of clustering over the five applications had similarities, and it was necessary to run many applications in order to get five sets of results where ten clusters were generated. More than 90% of the data representing single species was grouped into one cluster for between 6 and 8 out of 10 species. With between 6 and 9 out of 10 species containing more than 80% of the corresponding data in a single cluster. In all 5 applications, data representing *A. carterae*, *G. veneficum*, and *H. triquetra* were grouped together, all three species are from the same class. In 4 out of 5 cases, data representing *A. carterae*, *E. huxleyi*, and *N. pyriformis* were grouped together, they are from different classes.

In 3 out of 5 cases, data representing *A. carterae*, *A. pigmentosum* and *C. chiton* were grouped together, these species are in different classes, but all in the group taxon. (See Figure 4.33)

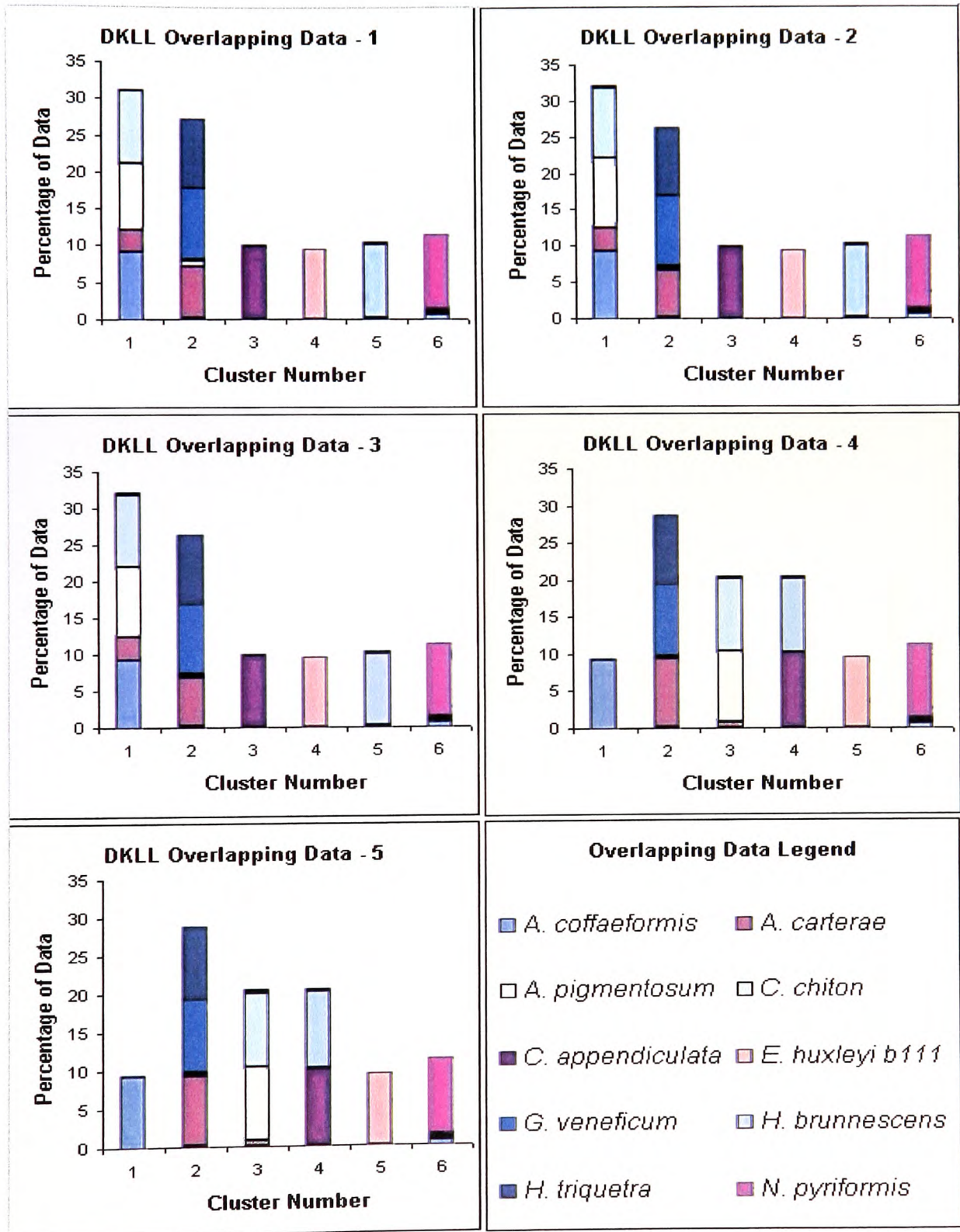
#### 4.7.1.2.3 Fourteen Clusters

Between ten and fourteen clusters were created by this algorithm over five applications. More than 90% of the data representing single species was grouped into one cluster for between 6 and 8 out of 10 species. With between 6 and 9 out of 10 species containing more than 80% of the corresponding data in a single cluster. In all 5 cases, data representing *C. chiton* and *N. pyriformis* were partitioned into the same cluster, these species are in different classes and not in the group taxon. In 4 out of 5 applications, data representing *A. carterae*, *G. veneficum*, and *H. triquetra* were grouped together, all three species are from the same class. In 3 out of 5 cases, data representing *A. carterae*, *A. pigmentosum* and *C. chiton* were grouped together, these species are in different classes, but all in the group taxon. (See Figure 4.34)

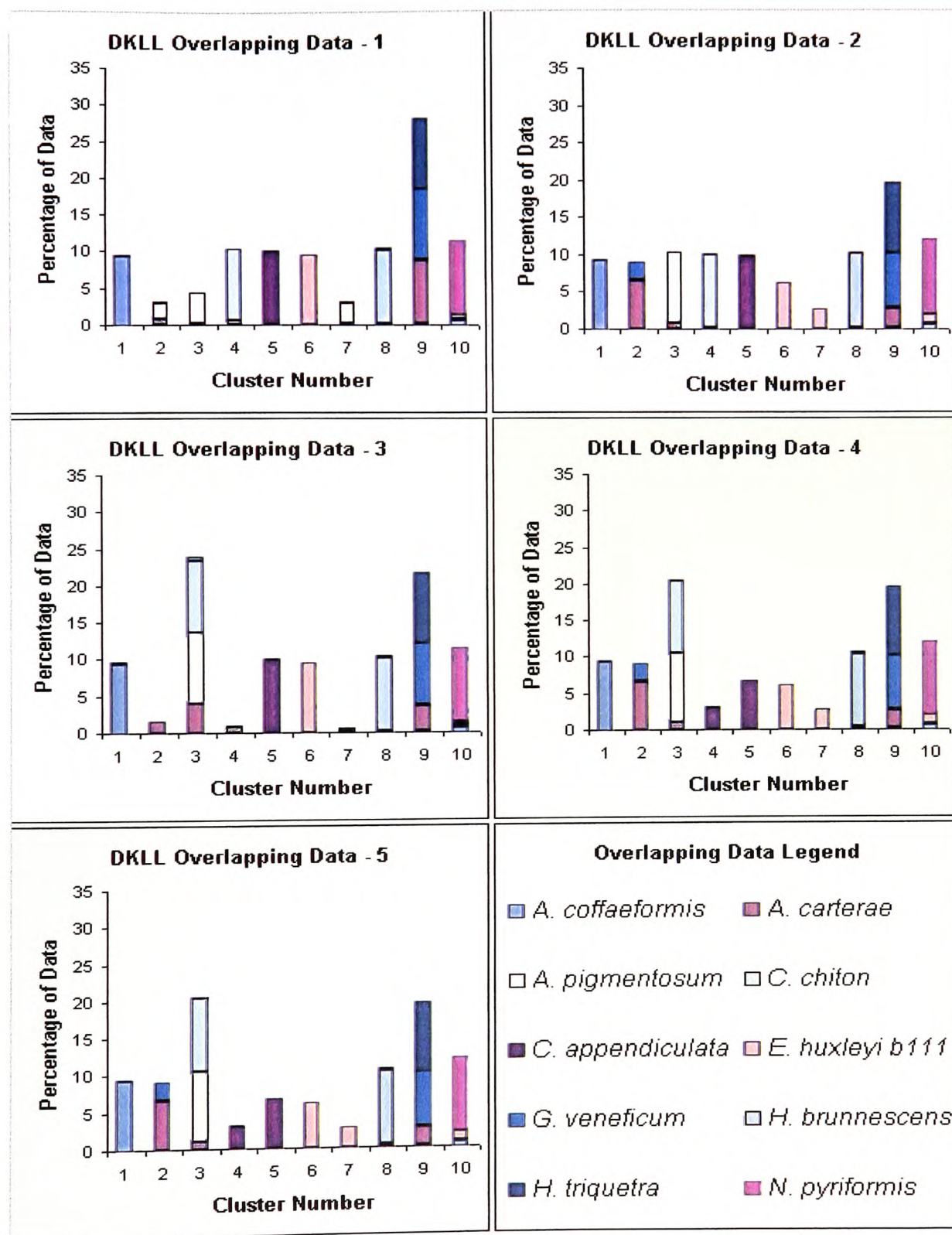
#### 4.7.1.2.4 Five Clusters – One Taxon

The data in the group taxon was partitioned into between 2 and 4 clusters in each application of the algorithm, between 97% and 98% of the group taxon data were partitioned in these clusters. In all 5 cases, data representing *C. appendiculata* and *H. brunnescens*, were grouped together, these species are in the same class. (See Figure 4.35)

**Figure 4.32 DKLL Overlapping Data - Six Clusters**

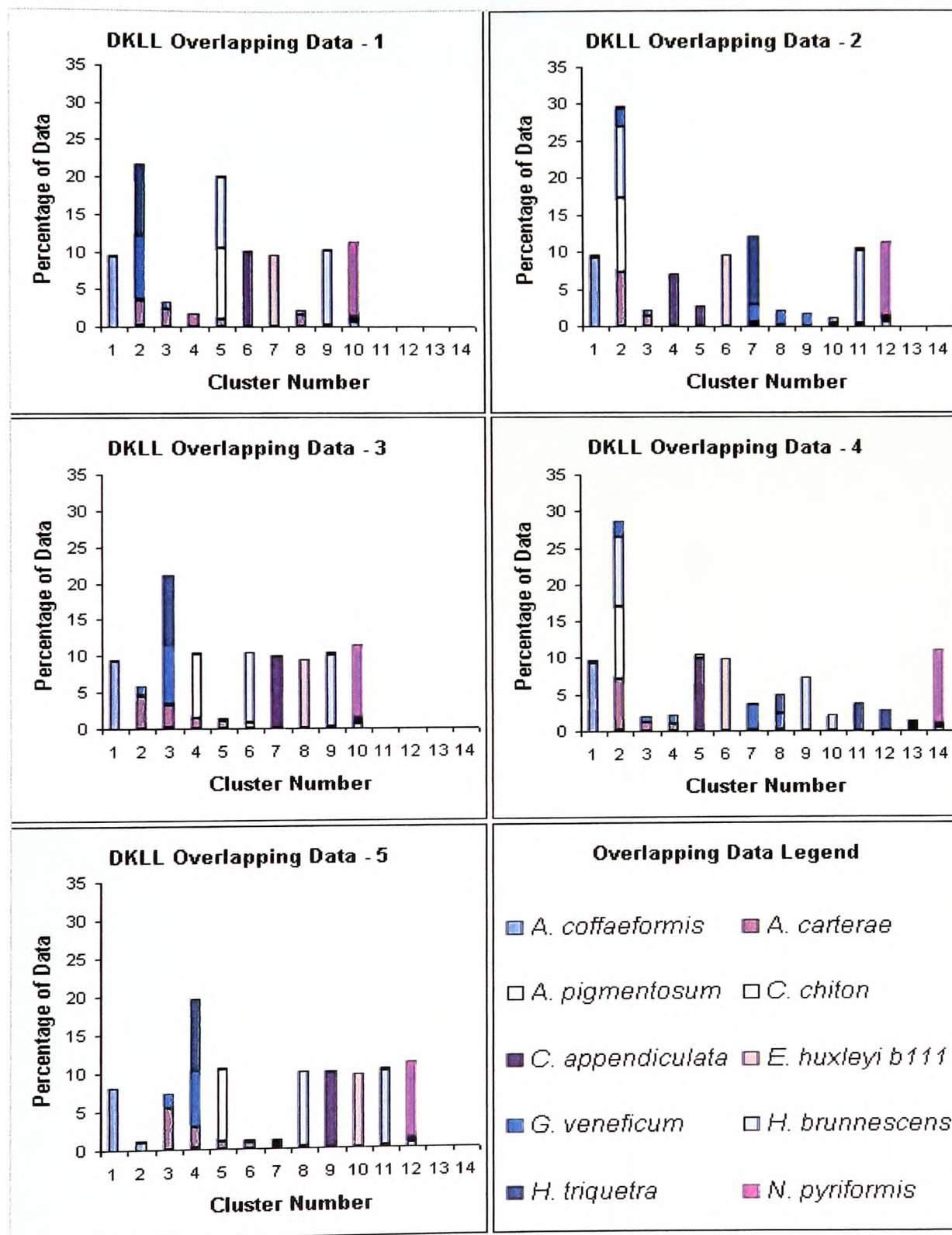


**Figure 4.33 DKLL Overlapping Data - Ten Clusters**

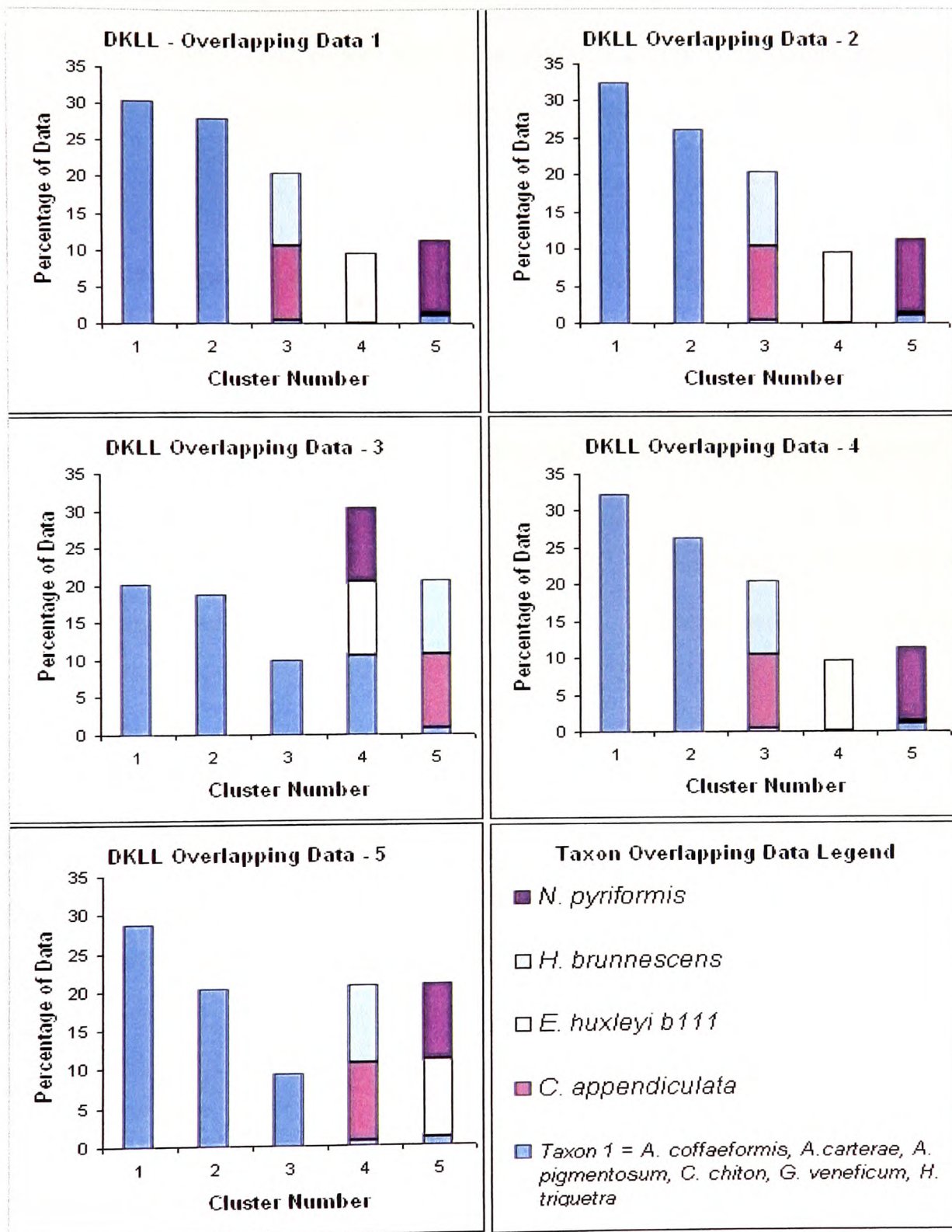




**Figure 4.34 DKLL Overlapping Data - Fourteen Clusters**



**Figure 4.35 DKLL Overlapping Data – Five Clusters / One Taxon**



## 4.8 Results Summary

The different runs of the algorithms produce different clustering solutions because the data points used as the initial cluster centres were chosen at random.

### 4.8.1 Comparison Of Algorithms Using Data With Non-Overlapping Clusters

**Table 4.1** Summary of times to convergence and consistency of clustering of the algorithms when using Data Set A (Non-Overlapping Clusters)

No. of clusters	Time to convergence	Consistency between replicates			Consistency of results for 10 clusters with "gold standard"
		6	10	14	
AD	68 secs	0.906	0.962	0.991	0.963
FKM	75 secs	0.907	0.954	*	0.940
DKLL	63 secs	0.921	0.947	0.965	0.948
ML	77 secs	0.857	0.890	0.929	0.843
MTV	*	*	*	*	*
SAND	86 secs	0.866	0.939	0.955	0.939

\* Did not converge to a solution

#### *4.8.1.1 Six Clusters*

Decreasing the number of clusters specified had the effect of combining some species within a cluster and splitting some species between clusters. DKLL allocated data patterns to clusters most consistently, although AD was only slightly less consistent. In contrast, increasing the specified number usually resulted in splitting species between clusters, with most clusters representing only a single species. AD allocated patterns most consistently followed by DKLL. MTV failed to converge to a solution. Further investigation showed that it tended to become trapped cycling between two clustering solutions. The other five clustering algorithms repeatedly converged when using six clusters.

#### *4.8.1.2 Ten Clusters*

With  $k=10$  clusters, AD produced clustering schemes in which each cluster was dominated by data patterns from a single species; it had the highest consistency between replicates and performed best in comparison to the gold standard (See Table 4.1). FKM resulted in a similar performance to AD, but 6 species tended to be paired together to get 3 clusters and 1 species was put in 2 clusters in significant proportions.

DKLL performed marginally less well, being less consistent between replicates and occasionally finding “composite” clusters containing two species, resulting in poorer performance in comparison to the gold standard. SAND performed slightly less well in terms of consistency between replicates, producing a higher proportion of composite clusters.



MTV failed to converge to a solution. ML frequently tended to produce a large aggregate cluster comprising patterns from eight or more species. This was a consequence of the unusual way in which it partitioned the data space. It generated small regions (representing the densely populated clusters) enclosed by a single large region (representing the remainder of the space). All the other algorithms produced partitions in which the regions were adjoining but continuous (i.e. in which no region was contained within another).

Table 4.1 also shows the times to convergence of each algorithm when using ten clusters. According to these times, DKLL has the fastest convergence and produces very consistent results, and AD has marginally the second fastest convergence but is even more consistent. This shows that the fastest converging algorithm does not necessarily produce the worst results and that more consistent results can be achieved by using a slightly slower algorithm. The algorithm which had the longest time to convergence is SAND which has variable consistency, showing that an algorithm with a longer convergence time does not necessarily produce the better results.

#### *4.8.1.3 Fourteen Clusters*

Again AD had the highest consistency between replicates and performed best in comparison to the gold standard. DKLL also produced consistently good results. (See Table 4.1) SAND performed better on the non-overlapping data, compared with ML, which repeatedly grouped together several discrete clusters

into one. FKM did not converge when applied to the non-overlapping set, and neither did MTV.

**4.8.2 Comparison Of Algorithms Using Data With Overlapping Clusters**

**Table 4.2** Summary of times to convergence and consistency of clustering of the algorithms when using Data Set B (Overlapping Clusters)

No. of clusters	Time to convergence	Consistency between replicates				Consistency of results for 10 clusters with "gold standard"
	10	5 (group)	6	10	14	10
AD	4 mins 49	0.807	0.878	0.971	0.973	0.946
FKM	11 mins 49	0.555	0.922	0.953	0.953	0.940
DKLL	3 mins 45	0.712	0.938	0.923	0.930	0.916
ML	5 mins 24	0.447	0.764	0.880	0.876	0.680
MTV	15 mins 32	*	*	0.954	*	0.940
SAND	20 mins 5	0.662	0.877	0.923	0.959	0.926

\* Did not converge to a solution

#### *4.8.2.1 Six Clusters*

FKM had the highest consistency when using six clusters. AD also performed consistently well, but not as well as FKM, and the same for DKLL. ML continued to repeatedly group together several discrete clusters into one. MTV did not converge, and SAND showed a tendency for the data to be partitioned so that four or five species were grouped in the same cluster.

#### *4.8.2.2 Ten Clusters*

All algorithms converged when ten clusters were specified, and all algorithms showed an increase in tendency to find “composite” clusters containing two or more species.

AD performed most consistently between replicates and with allocation of cells of known identity to the same cluster. Although SAND gave closer agreement with the five visually defined clusters than AD, overall it did not perform well with the overlapping data. FKM performed reasonably well, and MTV consistently failed to converge with the number of clusters specified that were not the natural number of clusters in the data set. ML again consistently partitioned the data so that a very large percentage of the data was grouped into one large cluster.

With the overlapping data, the time to convergence was significantly slower for all algorithms, however, the fastest algorithm was still DKLL which again had the second most consistent results, with AD slightly slower, but with better consistency, as with the non-overlapping data. (See Table 4.2)

#### *4.8.2.3 Fourteen Clusters*

FKM showed very good consistency, but partitioned data for a single species into more clusters than with non-overlapping data. AD had the highest consistency. ML continued to repeatedly group together several discrete clusters into one, and tended to only group the data into eleven or twelve clusters, not fourteen. MTV did not converge. SAND and DKLL also did not always partition the data into fourteen clusters, although DKLL had the second highest consistency. (See Table 4.2)

#### *4.8.2.4 Five Clusters – One Taxon*

For this case, the “gold standard” would be where there was one cluster containing Taxon 1, and the data for the other four species partitioned one species per cluster. FKM repeatedly grouped the data in taxon 1 into two or three clusters out of five and had a low rate of consistency. AD reliably grouped the data in taxon 1 into two main clusters, and had the highest consistency. SAND tended to group the data in taxon 1 into three clusters with one large and two much smaller clusters. MTV did not converge. When ML and DKLL were applied, the data was grouped differently each time with no real pattern to the clustering.

## 4.9 Conclusions

FKM performed consistently however it did not always converge, and SAND was more variable. ML was consistently poor and MTV was prone to non-convergence making these four algorithms unsuitable for the type of clustering application described here. AD performed the most consistently with both data sets, irrespective of the number of clusters specified, and DKLL was almost as good. However, with DKLL it was necessary to generate more than five results as the number of clusters the algorithm chose to use, varied slightly with the non-overlapping data, but quite significantly with the overlapping data. The algorithm was run more times to try to get five results where 10 clusters were produced. This was possible with the non-overlapping data, but not with the overlapping data set.

With AD, it consistently generated similar clustering solutions, regardless of the number of clusters specified *a priori*, or whether the data contained overlapping clusters or not. The 'Five Clusters / One Taxon' results show that between 95% and 98% of the taxon data was partitioned into 2 clusters, on each application of the algorithm. AD had the second fastest time to convergence, and had the highest consistency when ten and fourteen clusters were specified at 96% and 99% respectively, and the highest consistency of results for ten clusters with the 'Gold Standard' at 96%, when the non-overlapping data set was used. This method also had the second fastest time to convergence, and had the highest consistency when five, ten and fourteen clusters were specified at 81%, 97% and 97% respectively, and the highest consistency of results for ten clusters

with the 'Gold Standard' at 95%. Therefore it was concluded that the Adaptive Distances algorithm was the most suited to the clustering application studied.

Phytoplankton were often grouped together in the same cluster where sometimes the species were of the same group or class, but sometimes there was no taxonomic connection between the species. The data for two such species was plotted, and it could be seen that the main area of clustering was in the same place for both species. This may be due to the data being recorded at different times in the plankton lifecycles so that the species had similarities at that time. This means, that although it is possible to compare the algorithms to see which one clustered the species together by group or class in the best way, this will not provide a realistic view of how accurately the data of the species are being clustered. This may also be due to different species having the same flow cytometric signature. This is because the variables that are recorded are for classification purposes, e.g. are there any phytoplankton species in the water with red fluorescence, and are not for taxonomic identification.

In the present study, the number of clusters is specified beforehand, based, for example, on preliminary visual inspection of the data or on prior knowledge of the problem domain. Commonly, however, no prior information is available on the number of clusters in a data set. Techniques for automatically determining the number of clusters from the data have been developed using minimization of a cost function and/or persistence of clusters at different length scales [52] [39] [53].

## **Chapter 5 – Future Research**

### **5.1 Introduction**

In this research, it was necessary for the number of clusters in the data to be determined *a priori* when applying the clustering algorithms. Therefore one possible option for future research is to investigate algorithms which calculate the optimum number of clusters in unknown data sets as well as performing the classification.

Although some algorithms of this type exist, they have not been rigorously tested or applied to AFC data. AFC data can cause problems to clustering algorithms due to the elliptical nature, overlap of some clusters and biological variation of the species. The complexity of the data also means that some algorithms use approaches that are not currently computationally feasible. There are four possible methods that could be considered, namely:

- Minimising a regularised cost function (RCF) [53]
- Unsupervised Robust C Means (URCP) [37]
- The Robust Competitive Agglomeration Algorithm (RCA) [54]
- A Competitive Elliptical Clustering Algorithm (ECL) [55]

### **5.2 Minimising a Regularised Cost Function Algorithm (RCF)**

This algorithm purely determines the number of clusters in the data and the corresponding cluster centres, but it doesn't actually classify the data. It operates by using scale –space theory concepts in relation to clustering [53]. The



cost function consists of 2 parts; the first is the Euclidean distance metric, which calculates the cluster centres so that the sum-of-squared distance from each pattern to the cluster centre is minimised. The second part, is a regularisation term, which further dictates that the sum-of-squared distances between clusters is also minimised. The idea of this term is to draw cluster centres in the neighbourhood towards a 'winning' cluster which is the closest centre to a particular given input pattern. After training, clusters with a negligible winning count are deleted and the clusters within the neighbourhood of the 'winning' cluster are combined. The neighbourhood is shown to be a scale parameter, and the number of clusters is identified while varying the scale value. The number of clusters occurring most frequently over the largest range of the scale parameter is given as the number of clusters in the data, and the corresponding cluster centres calculated. If required, 'conscience learning' can be included in the algorithm to deter the same cluster centres from 'winning' too frequently. It is possible that different numbers of clusters can occur with the same high frequency over the same range of the scale parameter. In these cases, determining the 'best' solution can be based on either cluster validity or domain knowledge. In this algorithm, the Dunn index is applied as a measure of cluster validity and is used as a further criterion when choosing between possible solutions. This method is robust due to the implementation of the cluster validity, but would take a long time to converge unless the cluster centres were initialised to the "convergence values from the previous scale" [53]. Also, it was thought that the application of a scale parameter would result in an algorithm less

sensitive to noise. Once the cluster centres have been determined, they can then be superimposed on the data plots.

### **5.3 Unsupervised Robust C Means Algorithm (URCP)**

This clustering algorithm the Robust C Prototypes (RCP) algorithm is a combination of fuzzy clustering and statistical estimators and is based on the Fuzzy 'C' Means technique. Although the initial algorithm still requires the number of clusters to be known *a priori*, it was developed to become the Unsupervised Robust C Prototypes algorithms (URCP) to operate with an unknown number of clusters. This development is based on the initial number of clusters being greater than the actual number of clusters in the data, and therefore it is only necessary to know the maximum number of clusters that could be in the data. In this algorithm, two sets of weights (memberships) are generated for each data point. The first set partitions the data set, and the second memberships are unconstrained, and are used to determine robust estimates of the prototype parameters, which characterise each cluster. The evaluation of the prototype parameters can be severely affected by noise and outliers; this is because the constraint on the weights does not allow outlier or noisy data points "to have small membership values in all clusters simultaneously" [37]. To overcome the noise issue, the algorithm is developed to include M-estimators [56], and to minimise the effect of outliers, a loss function ( $\rho$ ) is associated with each cluster. The weight function argument is the squares of the distances and is always non-negative; consequently, a new

weight function was developed. It was based on the median of the distances, and the median of absolute deviations due to the robustness of these estimates. In order for ellipsoidal clusters to be identified, the Mahalanobis distance metric is used and the optimum number of clusters in unknown data is determined by repeatedly merging smaller clusters.

#### **5.4 A Robust Competitive Agglomeration Algorithm (RCA)**

Clustering algorithms can usually be categorised as either hierarchical or partitional clustering. The main advantages of hierarchical clustering are that, initialisation and the existence of local minima does not influence the clustering process, and that the number of clusters in the data does not need to be specified *a priori*. Prototype based clustering methods are the most popular of the partitional clustering category [57], and are classed as 'hard' or 'fuzzy' clustering. The advantages of these prototype approaches are that data points are able to move to different clusters in order to minimise the objective function, by using the relevant distance metric, information about the size and shape of the clusters can be combined within the objective function. In order to detect elliptical clusters, the Mahalanobis distance is used. The 'Competitive Agglomeration' (CA) is a technique which minimises a fuzzy prototype-based objective function iteratively which combines the advantages of both partitional and hierarchical clustering, and has the ability to identify many different shaped clusters. In the CA algorithm, the first step is to divide the data set into many small clusters. During each iteration, adjacent clusters 'compete' for data points, so that the

losing clusters tend to become smaller until they disappear. The final partition is said to have the optimum number of clusters in relation to the objective function. The CA technique is relatively robust to the effects of local minima and initialisation. The Robust Competitive Agglomeration (RCA) algorithm is an extension of CA in that it is also insensitive to noise and outliers. Concepts from robust statistics are combined in order for the RCA algorithm to be robust to noise. Here, for each data point, two sets of weights are assigned where, the first set partitions the data set, and the second is used to determine robust estimates of the “cluster prototypes” [54]. RCA is able to find an unknown number of clusters, recognise clusters of various shapes, and is robust to noisy data sets. The RCA algorithm is based on the Robust C Prototypes (RCP) algorithm used in the “Unsupervised Robust C Means” section featured previously.

### **5.5 A Competitive Elliptical Clustering Algorithm (ECL)**

Some clustering algorithms can have the problem where the covariance matrices become singular or nearly singular giving difficulties when inverting. This algorithm starts with spherical clusters and so avoids this singularity issue. It is based on Competitive Learning [58] which converges faster than some algorithms [55], and a frequency sensitive modification prevents larger clusters from being favoured. The classification part of the algorithm uses the Expectation Maximisation (EM) technique which has its roots in the Maximum Likelihood algorithm, and the update phase of the algorithm is described as sequential therefore the parameters are recalculated as each data point is

processed. Competitive Learning quickly tends to an optimum due to the small steps taken during this phase, and is less likely to result in local optima than the k-means algorithm [59]. For the case where elliptical clusters are found in data, covariance matrices represent the clusters and need to be updated. Therefore a covariance update rule is introduced to update the inverse of the matrices directly as opposed to after processing each data point. This algorithm is reported as producing good results even with overlapping data [55] .

## **5.6 Discussion of the Algorithms**

With the RCF algorithm, it is claimed to be able to find the optimum number of clusters, and seems to be able to identify several different types of data sets. For example, it successfully identified: non-overlapping clusters of varying spreads; overlapping clusters of varying spreads; and a combination of non-overlapping and overlapping clusters. As the data is known to contain clusters of varying spreads and overlaps, this is a highly desirable property. However, the data sets used were very small in comparison to ours and therefore it might not be a true reflection of its performance. It uses a 'cluster validity measure' when determining the optimum, which can be difficult to devise, and a single validity index is not likely to be appropriate for all clustering problems. Finally, this algorithm is based on the Euclidean distance metric. It would not be computationally feasible to use this distance metric with our data, and would also mean the algorithm would not be able to recognise any ellipsoidal clusters present in the AFC data.

The URCP algorithm is based on the RCP algorithm. It is able to find the optimum number of clusters, it uses the Mahalanobis metric to measuring the distance between points and clusters centres and is therefore able to recognise elliptical clusters, and it has been developed to overcome problems with noise and outliers. There is no reference made about whether overlapping data can be clustered and three variables are needed *a priori*, although if these variables were initialised to the 'wrong' values, this would only increase the time to convergence, it should not affect the accuracy of the results.

The RCA algorithm is also based on the RCP algorithm, and again uses two sets of weights, however the classification is achieved via different means. Once again, the Mahalanobis measure is used for ellipsoidal clusters, the algorithm is insensitive to noise and outliers as before, and is less sensitive to local minima and the effects of initialisation. Being insensitive to local minima makes this algorithm more likely to find the global optimum solution as opposed to a local one. Again there is no reference to the performance of the algorithm with overlapping data, and in this case, two parameters are needed *a priori* instead of three as previously. An equation is even given to estimate the maximum number of possible clusters, but again, the actual value assigned to these *a priori* parameters only affects the speed of the algorithm, and not the accuracy of the results.

As far as the ECL algorithm is concerned, the Mahalanobis distance metric is used which means the algorithm can identify elliptical clusters, and it uses spherical clustering initially to solve the problem of singular covariances.

The competitive learning aspect allows this algorithm to be adaptive to changing data, and is less susceptible to finding local optima than some algorithms. It incorporates a frequency sensitive approach to the competitive learning to prevent favouritism of the larger clusters, to have the ability to classify where different sized clusters are encountered, and is reported as producing good results even with some data overlap. Several parameters are required *a priori*, to initialise the mean positions of the clusters and the inter-class distance, no information is given about the acceptable range of values for these variables or the effects on the results. Unfortunately the classification phase is based on the Maximum Likelihood approach which produced some of the worst clustering results when classifying our AFC data sets.

After comparing the advantages and disadvantages of the prospective algorithms above, the two algorithms that are the most suited for this application are the Unsupervised Robust C Means Algorithm (URCP) and the Robust Competitive Agglomeration (RCA).

## **5.7 Other Options**

The RCF algorithm could be modified to use the Mahalanobis distance measure which can be easily computed, and which means the RCF algorithm would be capable of identifying elliptical clusters, which is essential for our data.

The ECL algorithm could be modified to use a different classification approach instead of the one based on Maximum Likelihood, e.g. Adaptive

Distances, which consistently produced good clustering solutions with both the overlapping and non-overlapping data in our data sets.

Also a 'hybrid' algorithm could be explored, either which would use the two 'new' algorithms simultaneously or two algorithms in series. Where the parallel approach is concerned, the results of the two techniques could be merged, by comparing each of the cluster centres in each algorithm, and by calculating a point which would effectively be the cluster centre of the possible cluster centre values. However, problems could arise if each method calculates a different number of clusters. In comparison, the two-stage approach could be a method where the 'modified' RCF algorithm could be used to determine the optimum number of clusters and the actual cluster centres. These cluster centres could then be input as the starting point for the RCA and / or URCP algorithm, which would classify the data.



## Bibliography

1. Wilkins, M.F., *Neural Network Analysis of Multivariate Flow Cytometric Data from Phytoplankton (PhD Thesis)*, in School of Biosciences. 1995, Cardiff University: Cardiff.
2. Wilkins, M.F., Hardy, S.A, Boddy, L, Morris, C.W., *Comparison of Five Clustering Algorithms to Classify Phytoplankton from Flow Cytometry Data*. *Cytometry*, 2001. **44**: p. 210-217.
3. Boney, A., *Phytoplankton*. 1975: Edward Arnold (publishers) Ltd.
4. Burkill, P.H., Mantoura, R.F.C., *The rapid analysis of single marine cells by flow cytometry*. *Philosophy Transactions of Royal Society*, 1990. **A333**: p. 99 - 112.
5. Grahame, J., *Plankton and Fisheries*. 1987: Edward Arnold (Publishers) Ltd, London.
6. Sournia, A., ed. *Phytoplankton Manual*. UNESCO monographs on ocean methodology. Vol. 6. 1978: Paris.
7. Burkill, P.H., *Analytical flow cytometry and its application to marine microbial ecology*. *Microbes in the Sea*, 1987: p. 133 - 166.
8. Dubelaar, G.B.J., Konig, J.W, Cunningham, A, Groenewegen, A.C, Jonker, R.R, Wietzorrek, J, Rutten, T.P.A, Beeker, A.E.R. *EurOPA: a novel 'high definition' flow cytometer for phytoplanktonic cells and colonies*. in *Oceans '94*. 1994. Osates, Brest.
9. Dubelaar, G.B.J., Jonker, R.R., *Flow Cytometry as a tool for the study of phytoplankton*. *Scientia Marina: Special issue on flow cytometry in aquatic science*, 2000.
10. Dubelaar, G.B.J., Groenewegen, A.C, Stokdijk, W, van den Engh, G.J, Visser, J.W.M, *Optical Plankton Analyser: A Flow Cytometer for Plankton Analysis. II: Specifications*. *Cytometry*, 1989. **10**: p. 529-539.
11. Dubelaar, G.B.J., Jonker, R.R, Meulemann, J.T.M, van Veen, J.J.F. *Phytoplankton Analysis by (EurOPA) Flow Cytometry; Current and Future Applications in Environmental Control*. in *Oceans '94*. 1994. Osates, Brest.
12. Carr, M.R., Tarran, G.A, Burkill, P.H, *Discrimination of marine phytoplankton species through the statistical analysis of their flow*

- cytometric signatures*. Journal of Plankton Research, 1996. **18**: p. 1225-1238.
13. Boddy, L., Morris, C.W, Wilkins, M.F, Tarran, G.A, Burkill, P.H., *Neural network analysis of flow cytometric data for 40 marine phytoplankton species*. Cytometry., 1994. **15**: p. 283-293.
  14. Johnson, R.A., Wichern, D.W, *Applied Multivariate Statistical Analysis*. 1992, New Jersey, USA: Prentice Hall International Inc.
  15. Kohonen, T., *An Introduction to Neural Networks*. Neural Networks, 1988. **1**: p. 3-16.
  16. Tou, J.T., Gonzalez, R.C, *Pattern Recognition Principles*. 1974, London: Addison-Wesley.
  17. Schalkoff, R.J., *Pattern Recognition: Statistical, Structural and Neural Approaches*. 1992, Chichester: Wiley International.
  18. Dayhoff, J.E., *Neural Network Architectures : An Introduction*. 1990, New York: Van Nostrand Reinhold.
  19. Lance, G.N., Williams, W.T., *A General Theory of Classificatory Sorting Strategies: II. Clustering Systems*. Computer Journal, 1967. **10**: p. 271-277.
  20. Jain, A., Dubes, K., Richard, C., *Algorithms for Clustering Data*. 1988: Prentice Hall.
  21. Balfourt, H.W., Snoek, J., Smits, J.R.M., Breedveld, L.W., Hofstraat, J.W., Ringelberg, J., *Automatic identification of algae: neural network analysis of flow cytometric data*. Journal of Plankton Research, 1992. **14**: p. 575-589.
  22. Frankel, D.S., Olson, R.J., Frankel, S.L., Chisholm, S.W., *Use of a neural net computer system for analysis of flow cytometric data of phytoplankton populations*. Cytometry, 1989. **10**: p. 540-550.
  23. Frankel, D.S., Frankel, S.L., Binder, B.J., Vogt, R.F., *Application of neural networks to flow cytometry data analysis and real-time cell classification*. Cytometry, 1996. **23**: p. 290-302.
  24. Wilkins, M.F., Boddy, L., Morris, C.W., Jonker, R.R., *Identification of Phytoplankton from Flow Cytometry Data by Using Radial Basis Function Neural Networks*. Applied and Environmental Microbiology, 1999. **65**(10): p. 4404 - 4410.

25. Boddy, L., Morris, C.W., Tarran, G.A., Burkill, P.H., Jonker, R.R., *Techniques for neural network identification of phytoplankton for the EUROPA flow cytometer*. Proceedings of OCEANS '94 OSATES Conference, 1994. **15**: p. 565-569.
26. Zadeh, L.A., *Fuzzy Sets*. Information and Control, 1965. **8**: p. 338-353.
27. Bellman, R.E., Kalaba, R.A., Zadeh, L.A., *Abstraction and Pattern Classification*. J.Math. Anal. Appl., 1966. **13**: p. 1-7.
28. Ruspini, E.H., *A new approach to clustering*. Inform. and Control., 1969. **15**: p. 22-32.
29. Duda, R.O., Hart, P.E., *Pattern Classification and Scene Analysis*. 1973, New York: Wiley.
30. Krishnapuram, R., Kim, J., *A note on the Gustafson-Kessel and adaptive fuzzy clustering algorithms*. IEEE Transactions On Fuzzy Systems, 1999. **7**(4): p. 453-461.
31. Gustafson, D.E., Kessel, W.C., *Fuzzy Clustering with a fuzzy covariance matrix*. Proc IEEE CDC, 1979. **2**: p. 761 - 766.
32. Trauwaert, E., Kaufman, L., Rousseeuw, P., *Fuzzy Clustering algorithms based on the Maximum Likelihood principle*. Fuzzy Sets and Systems, 1989. **42**: p. 213 - 227.
33. Trauwaert, E., Rousseeuw, P., Kaufman, L., *Fuzzy Clustering by Minimising the Total Hypervolume*. Information and Classification: Concepts Methods and Applications, 1993: p. 61 - 71.
34. Krishnapuram, R., Kim, J., *Clustering Algorithms Based on Volume Criteria*. IEEE Transactions On Fuzzy Systems, 2000. **8**(2): p. 228-236.
35. Demers, S., Kim, J, Legendre, P, Legendre, L., *Analysing multivariate flow cytometric data in aquatic sciences*. Cytometry, 1992. **13**: p. 291-298.
36. Rousseeuw, P., Kaufman, L, Trauwaert, E., *Fuzzy clustering using scatter matrices*. Computational Statistics and Data Analysis, 1996. **23**: p. 135 - 151.
37. Frigui, H., Krishnapuram, R., *A robust algorithm for automatic extraction of an unknown number of clusters from noisy data*. Pattern Recognition Letters, 1996. **7**(12): p. 1223 - 1232.

38. Boddy, L., Wilkins, M.F., Morris, C.W., *Pattern Recognition in Flow Cytometry*. Cytometry, 2001. **44**: p. 195-209.
39. Frattale Mascioli, F.M., Rizzi, A, Panella, M, Martinelli, G., *Scale-based approach to hierarchical fuzzy clustering*. Sig Proc, 2000. **80**: p. 1001-1016.
40. Sneath, P.H.A., *The application of computers to taxonomy*. Gen. Microbiology, 1957. **17**: p. 201-226.
41. Dunn, J., *A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters*. Journal of Cybernetics, 1974. **3**(3): p. 32 - 57.
42. Gao, X. and W. Xie, *Advances in theory and applications of fuzzy clustering*. Chinese Science Bulletin, 2000. **45**(11): p. 961-970.
43. Ramdas, V., Sridhar, V, Krishna, G., *An effective technique for feature extraction*. Pattern Recognition Letters, 1994. **15**: p. 885.
44. Bezdek, J.C., Castelaz, P.F., *Prototype classification and feature selection with fuzzy sets*. IEEE SMC, 1977. **2**(7): p. 87.
45. Boddy, L., Morris, C.W, Wilkins, M.F, Al-Haddad, L, Tarran, G.A, Jonker, R.R, Burkill, P.H., *Identification of 72 phytoplankton species by radial basis function neural network analysis of flow cytometric data*. Marine Ecology Progress Series, 2000 (reprint). **195**: p. 47 - 59.
46. Wilkins, M.F., Boddy, L, Morris, C.W. *AimsNet: Software for Artificial Neural Net Analysis of AFC Data*. in ISAC XX. 2000. Montpellier: Wiley-Liss.
47. Seamer, L.C., Bagwell, C.B., Barden, L., Redelman, D., Salzman, G.C., Wood, J.C.S., Murphy, R.F., *Proposed New Data File Standard for Flow Cytometry, Version FCS 3.0*. Cytometry, 1997. **28**: p. 118-122.
48. Bezdek, J., *Numerical Taxonomy with Fuzzy Sets*. Journal of Mathematical Biology, 1974. **1**: p. 57 - 71.
49. Bezdek, J., *Pattern Recognition with Fuzzy Objective Function Algorithms*. 1981: Plenum Press.
50. Rousseeuw, P., *Multivariate Estimation with High Breakdown Point*. Proc of the 4th Pannonian Symp on Math. Stat., 1983: p. 283 - 297.
51. Hartigan, J., *Clustering Algorithms*. 1975: John Wiley & Sons Inc.

52. Chakravarthy, S.V., Ghosh, J., *Scale-based clustering using the radial basis function network*. IEEE Trans Neural Networks, 1996. **7**: p. 1250-1261.
53. Kothari, R., Pitts, D., *On Finding the Number of Clusters*. Pattern Recognition Letters, 1999. **20**(4): p. 405-416.
54. Frigui, H., Krishnapuram, R., *A robust competitive clustering algorithm with applications in computer vision*. IEEE Transactions On Pattern Analysis and Machine Intelligence, 1999. **21**(5): p. 450-465.
55. De Backer, S., Scheunders, P., *A Competitive Elliptical Clustering Algorithm*. Pattern Recognition Letters, 1999. **20**(11 - 13): p. 1141 - 1147.
56. Huber, P., *Robust Statistics*. 1981: John Wiley and Sons.
57. Frigui, H., Krishnapuram, R., *Clustering by Competitive Agglomeration*. Pattern Recognition, 1997. **30**(7): p. 1109 - 1119.
58. Kohonen, T., *Self Organising Maps*. 2nd ed. 1997: Springer.
59. Scheunders, P., *A comparison of clustering algorithms applied to colour image quantization*. Pattern Recognition Letters, 1997. **18**: p. 1379 - 1384.

## **Appendices**

### **Appendix 1 – Glossary**

AD – Adaptive Distances

AFC – Analytical Flow Cytometry

ANN – Artificial Neural Network

DKLL – Demers, Kim, Legendre, Legendre

ECL – Elliptical Competitive Learning

FKM – Fuzzy K Means

MTV – Minimum Total Volume

RBF – Radial Basis Function

RCA – Robust Competitive Agglomeration

RCF – Regularised Cost Function

RCP – Robust C Prototypes

SAND – Sum of Normalised Determinants

URCP – Unsupervised Robust C Prototypes

RTAC – Real-Time Adaptive Clustering

## Appendix 2 – Web Site Directory

### Web 1 : Earth Observatory NASA website

<http://earthobservatory.nasa.gov/Library/Phytoplankton/phytoplankton2.htm>

c/o NASA Headquarters, Washington, DC 20546-0001(202) 358-0000

Website accessed 13<sup>th</sup> Oct 2003

### Web 2 : SeaWiFS website

[http://seawifs.gsfc.nasa.gov/SEAWIFS/sanctuary\\_4.html](http://seawifs.gsfc.nasa.gov/SEAWIFS/sanctuary_4.html)

c/o NASA Headquarters, Washington, DC 20546-0001(202) 358-0000

Website accessed 13<sup>th</sup> Oct 2003

### Web 3 : Plankton website

Photograph of *Nodularia spumigena* bloom, January 2002, in the Gippsland Lakes, Victoria, Australia. Photo credit: J.D. Kinnon

<http://www.whoi.edu/science/B/redtide/rtpotos/rtpotos.html>

Website accessed 13<sup>th</sup> Oct 2003

### Web 4 : Flow Cytometry information

<http://www.icnet.uk/axp/facs/davies/flow.html>

Website accessed 6<sup>th</sup> July 2000

### Web 5 : Neural Networks information

<http://www.cs.stir.ac.uk/~lss/NNIntro/InvSlides.html#algs>

### Web 6 : Clustering Algorithm information

[http://www.cne.gmu.edu/modules/dau/clustgalgs/clustgalgs\\_frm.html](http://www.cne.gmu.edu/modules/dau/clustgalgs/clustgalgs_frm.html)

### Web 7 : Maths and Stats information

<http://www.id.unizh.ch/software/unix/statmath/sas/sasdoc/stat/chap25/sect.htm>

Website accessed 13/11/03

### Web 8 : Principal Component Analysis information

[http://www.fon.hum.uva.nl/praat/manual/Principal\\_component\\_analysis.html](http://www.fon.hum.uva.nl/praat/manual/Principal_component_analysis.html)

Website accessed on 3/12/99

### Web 9 : Distance Metric information

[http://www.galactic.com/Algorithms/discrim\\_mahaldist.htm](http://www.galactic.com/Algorithms/discrim_mahaldist.htm)

Website accessed 2/12/99

### Web 10 : Mahalanobis Distance information

[http://www-engr.sjsu.edu/~knapp/HCIRODPR/PR\\_Mahal/PR\\_Mahal.htm](http://www-engr.sjsu.edu/~knapp/HCIRODPR/PR_Mahal/PR_Mahal.htm)

Website accessed 2/12/99

**Web 11 : Rhodophyceae information**

[www.ucmp.berkeley.edu/protista/rhodopyhta.html](http://www.ucmp.berkeley.edu/protista/rhodopyhta.html)

Website accessed 23<sup>rd</sup> July 2004

**Web 12 : Rhodophyceae information**

[www.ucmp.berkeley.edu/protista/reds/rhodolh.html](http://www.ucmp.berkeley.edu/protista/reds/rhodolh.html)

Website accessed 23<sup>rd</sup> July 2004

**Web 13 : Rhodophyceae information**

[www.ucmp.berkeley.edu/protista/reds/rhodomm.html](http://www.ucmp.berkeley.edu/protista/reds/rhodomm.html)

Website accessed 23<sup>rd</sup> July 2004

**Web 14 : Prymnesiophyceae information**

<http://www.ucmp.berkeley.edu/chromista/prymnesiophyta.html>

Website accessed 23<sup>rd</sup> July 2004

**Web 15 : Neural Network Information**

<http://www.aee-sf.org/editor.html>



## Appendix 3 – Conferences, Posters and Publications

### Conference Proceedings:

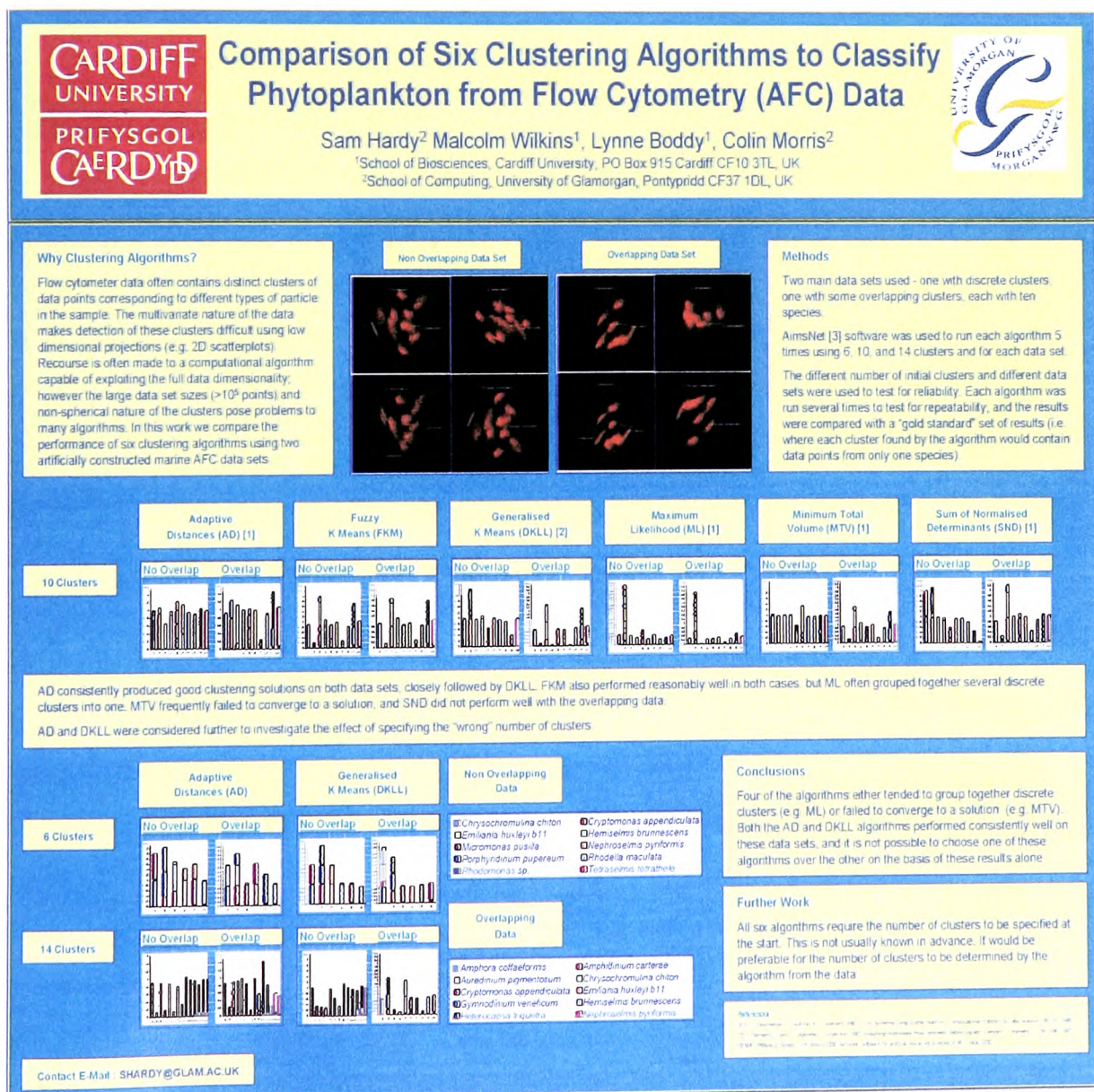
Poster:

Sam Hardy, Malcolm Wilkins, Lynne Boddy, Colin Morris

*Comparison of Six Clustering Algorithms to Classify Phytoplankton from Flow Cytometry (AFC) Data*

ISAC XX Conference 2000

Published in Cytometry Supplement 10 (2000)






**Posters:**

**Sam Hardy, Malcolm Wilkins, Lynne Boddy, Colin Morris**  
**Comparison of Six Clustering Algorithms to Classify Phytoplankton from Flow Cytometric Data**  
 Submitted to Pont Dysgu (2001), annual doctoral seminar at University of Glamorgan.

I won a Highly Commended certificate for this poster.




**CARDIFF UNIVERSITY**  
PRIFYSGOL CAERDYDD

## Comparison of Six Clustering Algorithms to Classify Phytoplankton from Flow Cytometry Data

Sam Hardy<sup>2</sup> Malcolm Wilkins<sup>1</sup>, Lynne Boddy<sup>1</sup>, Colin Morris<sup>2</sup>

<sup>1</sup>School of Biosciences, Cardiff University, PO Box 915 Cardiff CF10 3TL, UK  
<sup>2</sup>School of Computing, University of Glamorgan, Pontypridd CF37 1DL, UK



UNIVERSITY OF GLAMORGAN  
PRIFYSGOL MORGANŷ

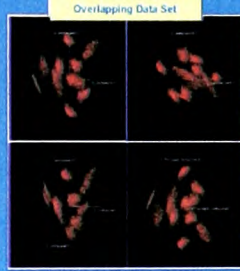
**Background**

Phytoplankton flow cytometric data often contains distinct clusters of data points corresponding to different species in the sample. The multivariate nature of this data makes detection of these clusters difficult using low dimensional projections (e.g. 2D scatterplots).

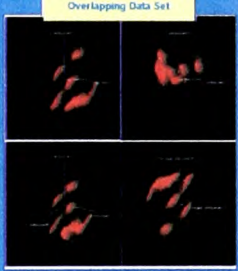
Alternatively, a computational algorithm capable of exploiting the full data dimensionality can be employed. Unfortunately the large data set sizes (>10<sup>3</sup> points) and non-spherical nature of the clusters limits which algorithms can be applied.

In this work the performance of six suitable clustering algorithms are compared using two artificially constructed data sets

2D Scatterplot of Non Overlapping Data Set



2D Scatterplot of Overlapping Data Set



**Methods**

Two main data sets used - one with discrete clusters, one with some overlapping clusters, each with ten species

AimsNet [3] software was used to run each algorithm 5 times using 6, 10, and 14 clusters and for each data set

The different number of initial clusters and different data sets were used to test for reliability. Each algorithm was run several times to test for repeatability, and the results were compared with a "gold standard" set of results (i.e. where each cluster found by the algorithm would contain data points from only one species).

Adaptive Distances (AD) [1]

Fuzzy K Means (FKM)

Generalised K Means (DKLL) [2]

Maximum Likelihood (ML) [1]

Minimum Total Volume (MTV) [1]

Sum of Normalised Determinants (SND) [1]

10 Clusters

No Overlap

Overlap

No Overlap

Overlap

No Overlap

Overlap

No Overlap

Overlap

No Overlap

Overlap

No Overlap

Overlap

No Overlap

Overlap

No Overlap

Overlap

Non Overlapping Data

- Chrysoschromulina chiton
- Emiliania huxleyi b11
- Micromonas pusilla
- Porphyridium papilliferum
- Rhodomonas sp.
- Cryptomonas appendiculata
- Hemiselmis brunneescens
- Nephroselmis pyriformis
- Thalassiosira weissflogii
- Thalassiosira weissflogii
- Thalassiosira weissflogii
- Thalassiosira weissflogii

**Results**

AD consistently produced good clustering solutions on both data sets, closely followed by DKLL. FKM also performed reasonably well in both cases, but ML often grouped together several discrete clusters into one. MTV frequently failed to converge to a solution, and SND did not perform well with the overlapping data.

AD and DKLL were considered further to investigate the effect of specifying the "wrong" number of clusters

Adaptive Distances (AD)

Generalised K Means (DKLL)

Overlapping Data

- Amphora coffaeiformis
- Chrysochromulina chiton
- Cryptomonas appendiculata
- Symmodium venustum
- Heterocapsa triquetra
- Amphionema carterae
- Chrysoschromulina chiton
- Emiliania huxleyi b11
- Hemiselmis brunneescens
- Nephroselmis pyriformis

**Conclusions**

Four of the algorithms either tended to group together discrete clusters (e.g. ML) or failed to converge to a solution (e.g. MTV). Both the AD and DKLL algorithms performed consistently well on these data sets, and it is not possible to choose one of these algorithms over the other on the basis of these results alone.

6 Clusters

14 Clusters

Contact Sam Hardy  
E-Mail : SHARDY@GLAM.AC.UK  
Phone : 01443 482312

Published  
Microscopical Journal of the British Microscopical Society, Volume 25, Issue 1, July 2006, pages 442-448, doi:10.1017/S0027217X06000042

Cardiff City  
Copyright © 2006 British Microscopical Society. All rights reserved. Printed in the United Kingdom. This journal is registered with the Copyright Clearance Center, 222 Rosewood Drive, Danvers, MA 01923, USA. Organizations in the USA who are also registered with the Copyright Clearance Center may therefore copy material (beyond the limits permitted by sections 107 and 108 of US copyright law) subject to payment to CCC of the per copy fee of \$12.00. This consent does not extend to multiple copying for promotional or commercial purposes. ISI Tear Sheet Service, 3501 Market Street, Philadelphia, PA 19104, USA, is authorised to supply single copies of separate articles for private use only. For all other use, permission should be sought from Cambridge or the American Branch of Cambridge University Press.

*Journal Publications:*

## Comparison of Five Clustering Algorithms to Classify Phytoplankton From Flow Cytometry Data

Malcolm F. Wilkins,<sup>1</sup> Sam A. Hardy,<sup>2</sup> Lynne Boddy,<sup>1\*</sup> and Colin W. Morris<sup>2</sup>

<sup>1</sup>Cardiff School of Biosciences, Cardiff, United Kingdom

<sup>2</sup>School of Computing, University of Glamorgan, Trefforest, Pontypridd, United Kingdom

Received 29 September 2000; Revision Received 8 March 2001; Accepted 12 April 2001

**Background:** Artificial neural networks (ANNs) have been shown to be valuable in the analysis of analytical flow cytometric (AFC) data in aquatic ecology. Automated extraction of clusters is an important first stage in deriving ANN training data from field samples, but AFC data pose a number of challenges for many types of clustering algorithm. The fuzzy k-means algorithm recently has been extended to address nonspherical clusters with the use of scatter matrices. Four variants were proposed, each optimizing a different measure of clustering "goodness."

**Methods:** With AFC data obtained from marine phytoplankton species in culture, the four fuzzy k-means algorithm variants were compared with each other and with another multivariate clustering algorithm based on critical distances currently used in flow cytometry.

**Results:** One of the algorithm variants (adaptive distances, also known as the Gustafson-Kessel algorithm) was found to be robust and reliable, whereas the others showed various problems.

**Conclusions:** The adaptive distances algorithm was superior in use to the clustering algorithms against which it was tested, but the problem of automatic determination of the number of clusters remains to be addressed. *Cytometry* 44: 210–217, 2001. © 2001 Wiley-Liss, Inc.

**Key terms:** automatic cluster extraction; clustering algorithms; phytoplankton; neural networks

Analytical flow cytometry (AFC) is a proven tool in aquatic ecology for the rapid analysis of water samples for detecting and quantifying microorganisms including phytoplankton and bacteria (1–3). The various light scatter, diffraction, and fluorescence parameters measured by AFC can provide characteristic "signatures" for each microbial cell, which allow taxa to be discriminated with the use of pattern-recognition techniques such as artificial neural networks (ANNs) (4–9). More than 70 species have been identified successfully by ANNs trained on AFC data obtained from pure cultures of marine microalgae grown under controlled conditions in the laboratory (6). However, species growing in the field are likely to show greater variability in size, shape, and pigmentation due to a multitude of environmental factors (10), thus producing a corresponding increase in the variability of the AFC signatures. ANNs require training on a representative sample of each species that is to be recognized, and, unless the training data reflect such biological variation, ANN analysis of field samples will not be reliable. There is therefore a need for a procedure for extracting ANN training data from field samples. The first step in such a method is to separate the AFC data obtained from field samples into their constituent clusters, e.g., for cell sorting with subsequent microscopic identification. Aside

from obtaining ANN training data, discerning clusters is the first step in the interpretation of AFC data.

The large data sets generated by AFC pose a number of challenges for analysis. The data sets are often multidimensional, making visual analysis by two-dimensional or three-dimensional scatter plots difficult, even with the use of dimensionality reduction techniques such as principal component analysis (11). Recourse is generally made to some form of clustering algorithm capable of exploiting the full multivariate nature of the data. However, the typical size of AFC data sets ( $>10^4$  patterns) precludes application of many standard clustering algorithms such as pair-group methods that rely on calculation of distances between pairs of points. AFC clusters are frequently highly elongated, and the variance-covariance structure can differ considerably between clusters. Some clusters are large and sparse, whereas others are compact and dense and perhaps even degenerate (possessing zero variance in one or more dimensions and therefore zero vol-

Grant sponsor: AIMS, Commission of the European Community; Grant number: (CEC) MAS3-CT97-0080.

\*Correspondence to: Lynne Boddy, Cardiff School of Biosciences, P.O. Box 915, Park Place, Cardiff CF10 3TL, UK.

E-mail: boddy@cardiff.ac.uk

ume). Demers et al. (12) proposed an iterative cluster analysis algorithm for AFC data capable of modeling the heterogeneous nature of typical AFC clusters, but it was not always successful (see below). The current study compares the performance of the algorithm of Demers et al. (12) with four others (apparently not previously applied to AFC data) and with manual cluster extraction using two-dimensional and three-dimensional scatter plots with the use of artificially constructed marine phytoplankton AFC data sets.

### THE ALGORITHMS

The algorithm of Demers et al. (12) (Appendix 1), henceforth denoted as DKLL, was based on an extension of the classic k-means algorithm (13) to allow for nonspherical clusters through the use of variance-covariance matrices (scatter matrices) to model the data distribution around each cluster center. A common problem with such iterative algorithms is that, over the course of several iterations, one cluster tends to expand to include all the data at the expense of the other clusters. The DKLL algorithm incorporated the use of threshold "critical distances" to prevent this. At each step of the algorithm, only those points within the critical distance of the cluster center were used in calculating the corresponding scatter matrix. The critical distances were increased gradually during successive iterations under the assumption that the clustering progressively would become more reliable. In practice, however, that approach was not ideal: there are no guidelines as to how rapidly the critical distances should be increased, and the number of data points within the critical distance of the cluster center may well fall to zero during the course of the algorithm, leaving the cluster scatter matrix undefined. Furthermore, the algorithm frequently fails to converge to an optimal clustering solution, requiring user intervention to reject inadequate results (12).

Fuzzy k-means clustering (14,15) might overcome these problems. It is a variant of classic k-means clustering that allows data points to become associated to some degree with all clusters and not just the closest cluster, thereby reflecting the inherent uncertainty in allocating a data point to a single cluster where there are several potential candidates. The algorithm is in practice stable and robust and less likely than the classic k-means algorithm to produce inadequate clustering solutions. The algorithm has been extended by different investigators to include the use of scatter matrices, thereby allowing modeling of data with nonspherical clusters. Rousseeuw et al. (16) unified that work by proposing four variants of a generic fuzzy k-means algorithm: adaptive distances (AD), maximum likelihood (ML), minimum total volume (MTV), and sum of all normalized determinants (SAND; Appendix 2). Each variant was designed to minimize a different objective criterion (measure of clustering "goodness"). The AD algorithm, also known as the Gustafson-Kessel algorithm (16), seeks to minimize the fuzzy sum of squared generalized distances of the data patterns to the cluster centers, subject to the constraint that the determinant of the scat-

ter matrices (a measure of cluster volume) can be fixed in advance. The ML algorithm assumes that each cluster represents a multivariate normal probability distribution and attempts to find a clustering solution that maximizes the overall likelihood of the data set over all clusters and data items; this method tends to seek cluster solutions where all clusters have similar volumes (16). The MTV algorithm minimizes the total cluster volume and is biased toward finding clustering solutions where the clusters have similar densities rather than similar volumes. The SAND algorithm attempts to reduce this bias by normalizing the cluster volumes by dimensionality.

### MATERIALS AND METHODS

#### Data Sets

A large set of 7-parameter FACScan flow cytometric data for 63 marine phytoplankton species, each grown in pure culture (6), was used. The data sets were examined visually using three-dimensional principal component analysis scatter plots to determine which species had data distributions that overlapped and which did not. Two artificial data sets, A and B, respectively, each of 10 species, were constructed from the plots to produce a data set comprising species for which the corresponding AFC clusters did not overlap appreciably (Fig. 1a) and a data set comprising species for which the corresponding AFC clusters overlapped considerably (Fig. 1b). Data sets were constructed by randomly selecting  $10^3$  data patterns for each of these species. Both data sets thus contained a total of  $10^4$  data patterns.

#### Comparing the Performances of the Five Algorithms

Each algorithm was run five times on the nonoverlapping (A) and overlapping (B) data sets for  $k = 6$ ,  $k = 10$ , and  $k = 14$  clusters. Two criteria were used for evaluating performance: (i) consistency of results between multiple replicate runs of the algorithm for the same value of  $k$  and (ii) performance of the algorithm in comparison with a "gold standard" clustering scheme (corresponding to the actual identity of the data patterns, which for this problem was known in advance). Consistency between multiple algorithm runs was assessed by averaging a similarity measure (see next section) between clustering schemes from pairs of replicates. Consistency with the gold standard was assessed by averaging the similarity measure between the clustering schemes from five replicate results obtained with  $k = 10$  and the gold standard clustering scheme. For the nonoverlapping data set, each algorithm was evaluated with both criteria; for the overlapping data set, only criterion (i) was used, because the gold standard is taken as the known identity of a particular cluster: for the nonoverlapping data, each species represents a distinct cluster, but this is not the case for the overlapping data set. By using traditional taxonomic characters, the 10 species in data set B would form distinct clusters, but this was not the case with the seven flow cytometric parameters measured in the present study, which resulted in



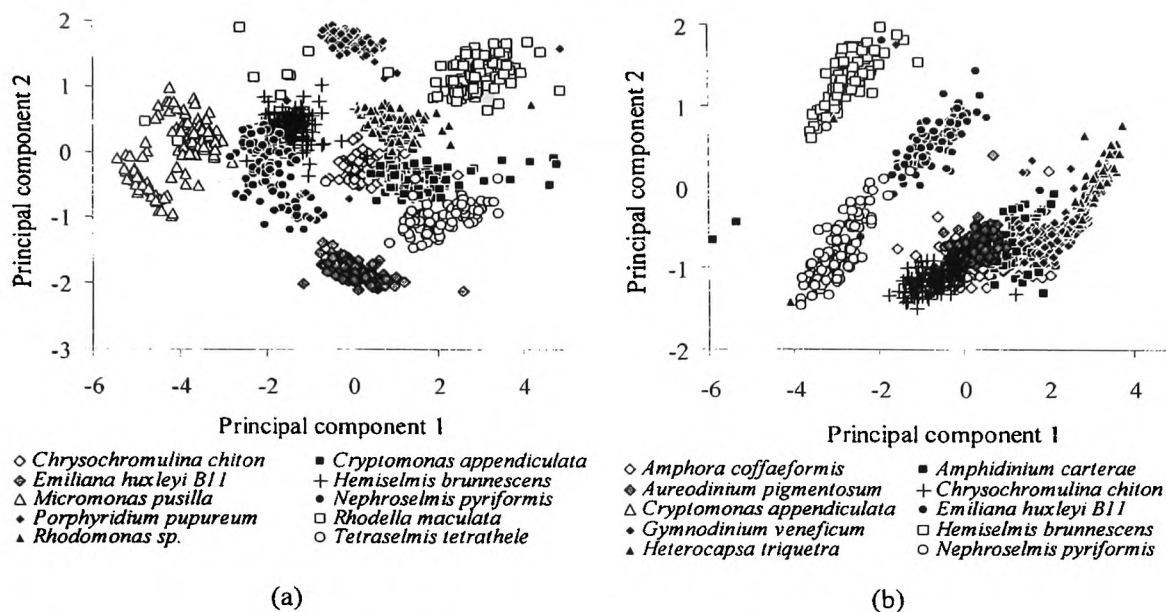


FIG. 1. Plots illustrating the distributions of the 10 species in data sets A (a; nonoverlapping clusters) and B (b; overlapping clusters) projected in each case onto the plane of the first two principal components. For data set A, all 10 clusters are clearly resolved in three-dimensional plots of principal components analysis (not presented).

overlapping distributions. Hence, in the case of data set B, we do not know what the "gold standard" should be.

#### A Similarity Measure for Comparing the Results of Two Clusterings

A good clustering algorithm should produce results that are similar to the clusters that can be noted visually in the data (where these are clearly defined) and are consistent, i.e., similar over several replicate runs of the algorithm on the same data. The result of running a clustering algorithm on a set of data is a partitioning of the data space into regions; applying this partition to the data set (by assigning each data point an identity based on the region in which it resides) results in a "clustering scheme" for the data. A clustering scheme also can be defined manually by visual inspection of the data with the use of scatter plots. An objective method is thus required for quantifying the similarity between two clustering schemes. Given two clustering schemes, C and D, the proposed measure of similarity between them,  $m(C,D)$ , is defined as the a priori probability that two randomly selected points drawn from the data set will be clustered in the same way under both clustering schemes; i.e., that both clustering schemes will assign both points to the same cluster or both clustering schemes will assign them to separate clusters. By letting  $c$  be the event "both points are assigned to the same cluster under C" and  $d$  be the event "both points are assigned to the same cluster under D," we can write  $m(C,D) = p(c)p(d|c) + p(c')p(d'|c')$ . The characteristics of this similarity measure are: (i) if C and D are identical (perfect correlation),  $p(d|c) = p(d'|c') = 1.0$ , thus  $m(C,D) = 1.0$ ; (ii) if C and D are independent (no correlation),  $p(d|c) =$

$p(d)$  and  $p(d'|c') = p(d')$ , thus  $m(C,D) = p(c)p(d) + p(c')p(d')$ ; and (iii) it is commutative; i.e.,  $m(C,D) = m(D,C)$ .

Because of the large size of typical AFC data sets, determining  $m(C,D)$  by exhaustive enumeration of all possible pairs of data points is computationally unfeasible; however, the proportion can be estimated to a desired level of accuracy, and confidence bounds placed on the value are obtained by repeatedly, randomly drawing a large number  $n$  of pairs of data points and finding the proportion  $p$  of such trials for which the two schemes agree. The sampling standard deviation of  $p$ , a measure of the size of the uncertainty in the value of  $p$  as an estimator of  $m(C,D)$ , is given by  $q = (np[1 - p])^{1/2}/n = (p[1 - p]/n)^{1/2}$ . All similarity measures reported in this paper were evaluated from  $n = 10^7$  pairs of data points.

#### Computer Software

All experiments used AimsNet software running on a PC under Microsoft Windows NT (<http://www.cf.ac.uk/biosi/staff/wilkins/aimsnet>), a computer package developed as part of AIMS, a CEC-funded project (grant MAS3-CT97-0080).

## RESULTS AND DISCUSSION

### Data Set A (Nonoverlapping Clusters)

With  $k = 10$  clusters, AD produced clustering schemes in which each cluster was dominated by data patterns from a single species (Fig. 2); it had the highest consistency (i.e., lowest variability) between replicates and performed best in comparison with the gold standard (Table

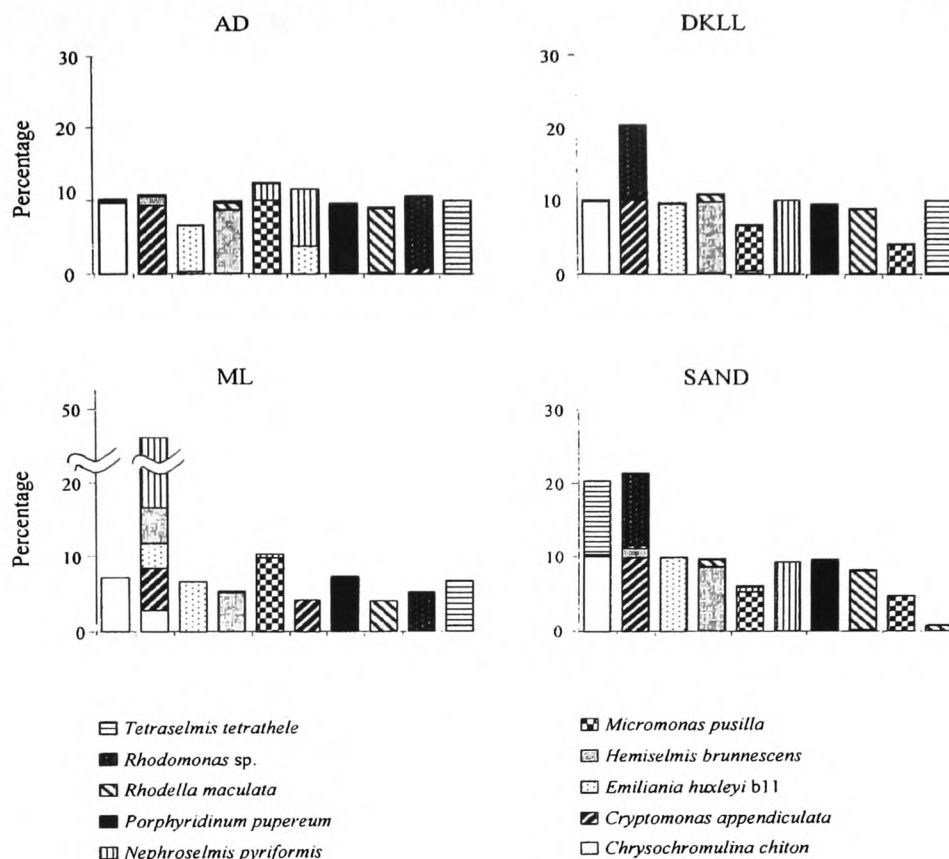


FIG. 2. Typical partitioning of data set A (nonoverlapping clusters) between 10 clusters by algorithms for adaptive distances (AD), Demers (DKLL), sum of all normalized determinants (SAND), and maximum likelihood (ML). The bars show the percentage of the data set associated with each cluster and the composition of the data associated with each cluster in terms of the 10 constituent species.

1). DKLL performed marginally less well, being less consistent between replicates and occasionally finding "composite" clusters containing two species, resulting in poorer performance in comparison with the gold standard. SAND performed slightly less well in terms of consistency between replicates, producing a higher propor-

Table 1  
Comparison of Algorithms for Data Set A  
(Nonoverlapping Clusters)

Number of clusters	Consistency between replicates <sup>a</sup>			Consistency of results compared with "gold standard" <sup>b</sup>
	6	10	14	
AD <sup>b</sup>	0.906	0.962	0.991	0.963
DKLL	0.921	0.947	0.965	0.948
ML	0.857	0.890	0.929	0.843
MTV	<sup>c</sup>	<sup>c</sup>	<sup>c</sup>	<sup>c</sup>
SAND	0.866	0.939	0.955	0.939

<sup>a</sup>Proportion of trials for which two schemes agree. Values range between 0 and 1.0 for no agreement to complete agreement. Standard deviation of the proportion can be derived from the equation given in the text.

<sup>b</sup>AD, adaptive distances; DKLL, from Demers et al. (12); ML, maximum likelihood; MTV, minimum total volume; SAND, sum of all normalized determinants.

<sup>c</sup>Did not converge to a solution.

tion of composite clusters. MTV frequently failed to converge to a solution, particularly when the number of clusters specified differed from the optimal, or "natural," number of clusters (further investigation showed that it tended to become trapped cycling between two clustering solutions). ML frequently produced a large aggregate cluster comprising patterns from eight or more species, resulting from the unusual way in which it partitioned the data space: it tended to generate small regions (representing the densely populated clusters) enclosed by a single large region (representing the remainder of the space). All the other algorithms produced partitions in which the regions were adjoining but continuous (i.e., in which no region was contained within another).

The effect of decreasing and increasing the specified number of clusters on the clustering solutions was investigated. Decreasing the number of clusters specified had the effect of combining some species within a cluster and splitting some species between clusters (Fig. 3). Underspecifying the number of clusters in comparison with the "natural" number resulted in the results of the AD algorithm becoming less consistent (i.e., more variable) between replicates than DKLL (Table 1). Overspecifying the number of clusters usually resulted in splitting species between clusters, with most clusters representing only a single species; under those circumstances, the results of

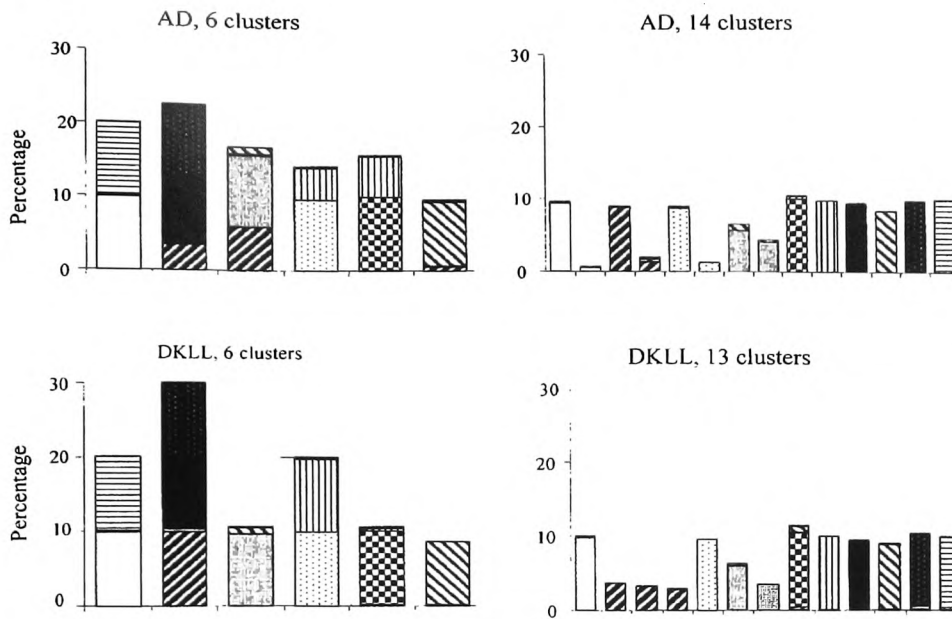


FIG. 3. Typical partitioning of data set A (nonoverlapping clusters) between 6 and 14 clusters by algorithms for adaptive distances (AD) and Demers (DKLL); symbols and data as in Figure 2.

the AD algorithm were much more consistent between replicates than DKLL.

**Data Set B (Overlapping Clusters)**

All algorithms converged when 10 clusters were specified, although MTV failed to do so when other numbers of clusters were specified. AD performed significantly more consistently than the DKLL or SAND for  $k = 10$  and  $k = 14$ , although not for  $k = 6$  (Table 2). The results for SAND (Fig. 4) appeared to produce closer agreement with the visually defined clusters (Fig. 1b).

**Defining the Number of Clusters**

In the present study, the number of clusters  $k$  was specified beforehand, based, e.g., on preliminary visual inspection of the data or prior knowledge of the problem domain. AD performed the most consistently with both data sets, provided that the number of clusters specified

was the same as or greater than the actual number of clusters present in the data. DKLL was almost as good, and both are worth further investigation. SAND was more variable, ML was consistently poor, and MTV was prone to nonconvergence, making these algorithms unsuitable for the type of clustering application described in this study.

In general, no prior information is available on the number of clusters in a data set, and trying to decide on the number of clusters is the classic problem in cluster analysis (e.g., 17). The view has been taken that number of clusters has to be decided subjectively and that this necessitates the involvement of a biologist expert in the field. A trial-and-error approach often is adopted to depict the data structure appropriately (e.g., 18). Because computers have become fast and efficient, this is not a serious disadvantage because a wide range of potential numbers of clusters can be tried. It can be cogently argued that nonhierarchical clustering approaches do not stand well on their own and, in many cases, should be used with other complementary methods of exploratory data analysis, which will give further insight into numbers of clusters (18).

Many techniques for automatically determining the number of clusters from the data have been proposed, with most of the early ones incorporating sums of squares and having a geometric or statistical interpretation (19). Interesting, recent developments have used minimization of a cost function and/or persistence of clusters at different length scales (20-22). However, that method may not yield the number of clusters expected or noted visually.

There are some clustering approaches that do not require the number of clusters to be specified in advance, e.g., density clustering (23) and adaptive clustering (24). Density clustering is a nonparametric, noniterative statistical method that essentially examines the local event

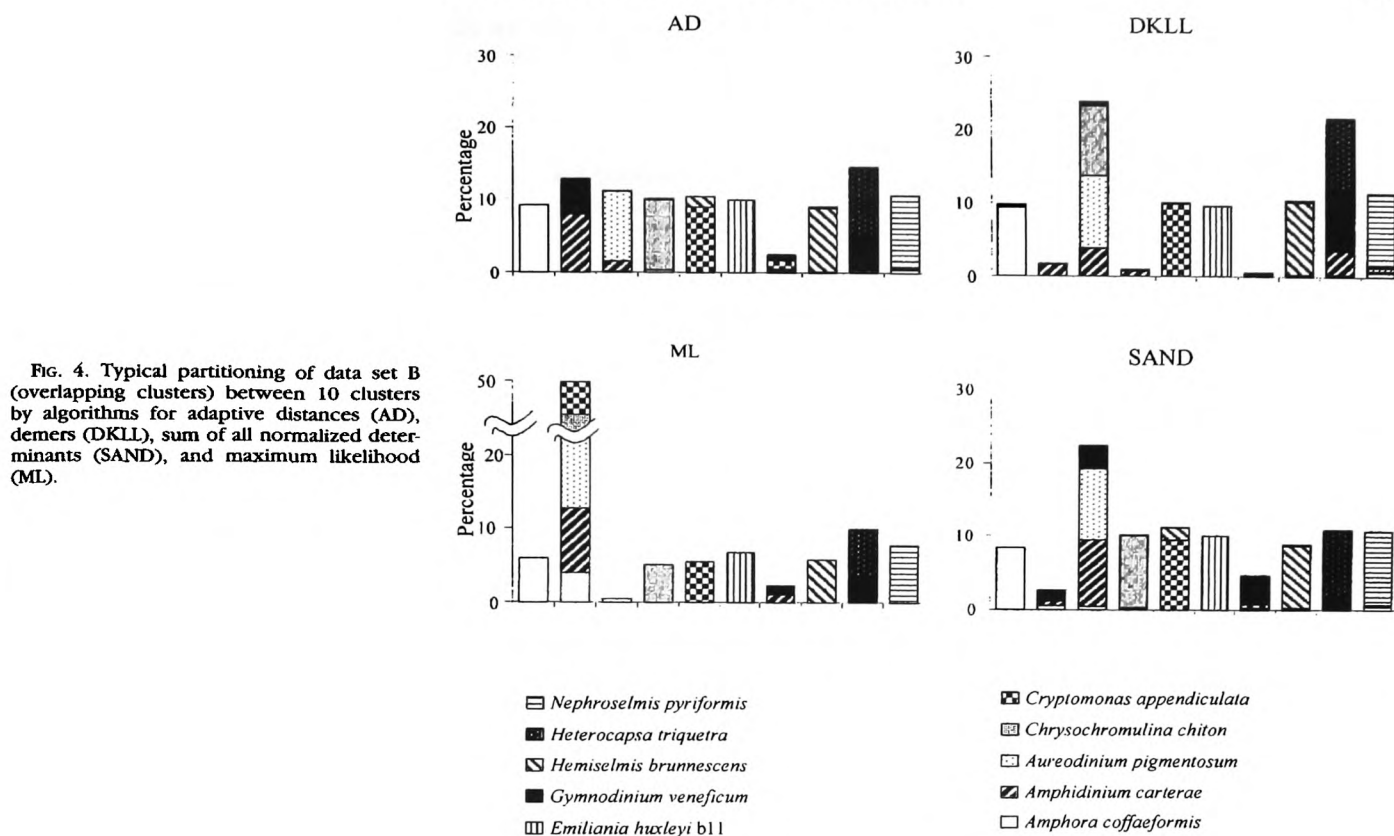
Table 2  
Comparison of Algorithms for Data Set B  
(Overlapping Clusters)

Number of clusters	Consistency between replicates <sup>a</sup>		
	6	10	14
AD <sup>b</sup>	0.878	0.971	0.973
DKLL	0.938	0.923	0.930
ML	0.764	0.880	0.876
MTV	<sup>c</sup>	0.954	<sup>c</sup>
SAND	0.877	0.923	0.959

<sup>a</sup>As in Table 1.

<sup>b</sup>AD, adaptive distances; DKLL, from Demers et al. (12); ML, maximum likelihood; MTV, minimum total volume; SAND, sum of all normalized determinants.

<sup>c</sup>Did not converge to a solution.



density (based on histogram counts) to determine modes, and then regions of monotonically decreasing density around each of these are grouped into a cluster (23). However, a user-specified threshold determines the size of clusters, which is subjective. Real-time adaptive clustering is an ANN approach in which the data pattern for each cell is presented to the network and then allocated to a cluster of similar patterns (24). When a pattern is not similar to any existing cluster, it is allocated to a new cluster. The ANN is trained to classify events according to a predefined similarity measure, but this similarity measure is subjectively determined. Thus, although these approaches have the advantage of not requiring the number of clusters to be specified in advance, they are not entirely objective.

#### CONCLUSIONS AND FUTURE RESEARCH

The AD algorithm was superior in use to the algorithms against which it was tested. Moreover, it is widely applicable and likely to be of use with other very large data sets, such as might be produced by other high-technology equipment with rapid processing capabilities. This is clearly a big step forward but its potential cannot be fully realized until a suitable objective method for determining the number of clusters in a data set has been developed, to allow completely automatic clustering. Nonetheless, even if subjective decisions (with or without the help of exploratory data analysis methods) are made about the num-

ber of clusters, using the AD algorithm to extract data sets that can subsequently be used for training ANNs will be extremely valuable. Ultimately, clustering software could be used to drive cytometer cell-sorting facilities, which will enable the operator to examine cells clustered together on the basis of their flow cytometric signatures with the use of other biological or physicochemical analyses. Cell sorting also would allow the user to check whether clusters contain single or several cell types. Therefore, it would be invaluable if the next generation of flow cytometers has the ability to interface custom-written software with, e.g., the cell-sorting facility.

In future research, it would be valuable to compare AD with other clustering approaches that have been used with flow cytometric data including density clustering (23), real-time adaptive clustering (24), and Kohonen self-organizing maps (SOMs) (25-27). The first two approaches are important because they do not require specification of the number of clusters. SOMs are worth comparing because they result in a low-dimensional, topology-preserving representation of the data set. They are an ANN approach in which high-dimensional data are mapped onto a two-dimensional (usually) lattice of nodes, each of which competes to represent the data presented to the network. Patterns represented by nodes that are close to each other in the lattice typically are similar to each other. This approach is alluring although methods of



separating clusters have not been fully developed, and there is still the problem of deciding on the number of clusters.

#### ACKNOWLEDGMENTS

AimsNet was developed during AIMS, a project funded by the Commission of the European Community, CEC grant MAS3-CT97-0080. Thanks are due to Dr. G. A. Tarran for provision of the AFC data on which this work was based, collected as part of a Natural Environment Research Council PRiME Special Topic Award (GST/02/1062).

#### LITERATURE CITED

- Burkill PH. Analytical flow cytometry and its application to marine microbial ecology. In: Sleigh MA, editor. *Microbes in the sea*. Chichester: Horwood; 1987. p 139-166.
- Burkill PH, Mantoura RFC. The rapid analysis of single marine cells by flow cytometry. *Phil Trans R Soc* 1990;A333:99-112.
- Hofstraat JW, Van Zeijl WJM, De Vreeze MEJ, Peeters JCH, Peperzak L, Colijn F, Rademaker TWM. Phytoplankton monitoring by flow-cytometry. *J Plankton Res* 1994;16:1197-1224.
- Balfort HW, Snoek J, Smits JRM, Breedveld LW, Hofstraat JW, Ringelberg J. Automatic identification of algae: neural network analysis of flow cytometric data. *J Plankton Res* 1992;14:575-589.
- Boddy L, Morris CW, Wilkins MF, Tarran GA, Burkill PH. Neural network analysis of flow cytometric data for 40 marine phytoplankton species. *Cytometry* 1994;15:283-293.
- Boddy L, Morris CW, Wilkins MF, Al-Haddad L, Tarran GA, Jonker RR, Burkill PH. Identification of 72 phytoplankton species by radial basis function neural network analysis of flow cytometric data. *Mar Ecol Prog Ser* 2000;195:47-59.
- Frankel DS, Olson RJ, Frankel SL, Chisholm SW. Use of a neural net computer system for analysis of flow cytometric data of phytoplankton populations. *Cytometry* 1989;10:540-550.
- Frankel DS, Frankel SL, Binder BJ, Vogt RF. Application of neural networks to flow cytometry data analysis and real-time cell classification. *Cytometry* 1996;23:290-302.
- Wilkins MF, Boddy L, Morris CW, Jonker R. Identification of phytoplankton from flow cytometry data using radial basis function neural networks. *Appl Environ Microbiol* 1999;65:4404-4410.
- Boddy L, Wilkins MF, Morris CW, Tarran GA, Burkill PH, Jonker RR. Techniques for neural network identification of phytoplankton for the EurOPA flow cytometer. In: *Conference proceedings of Oceans 94 OSATES, Brest, 1994*.
- Jolliffe IT. *Principal components analysis*. Springer series in statistics. New York: Springer-Verlag; 1986.
- Demers S, Kim J, Legendre P, Legendre L. Analysing multivariate flow cytometric data in aquatic sciences. *Cytometry* 1992;13:291-298.
- Hartigan J. *Clustering algorithms*. New York: Wiley; 1975.
- Bezdek JC. *Numerical taxonomy with fuzzy sets*. *J Math Biol* 1974;1:57-71.
- Bezdek JC. *Pattern recognition with fuzzy objective function algorithms*. New York: Plenum Press; 1981.
- Rousseeuw PJ, Kaufman L, Trauwaert E. Fuzzy clustering using scatter matrices. *Comput Stat Data Anal* 1996;23:135-151.
- Dale MB. Knowing when to stop: cluster concept-concept cluster. *Coenoses* 1988;3:11-32.
- Podani J. *Introduction to the exploration of multivariate biological data*. Leiden, The Netherlands: Backhuys Publishers; 2000.
- Milligan GW, Cooper MC. An examination of procedures for determining the number of clusters in a data set. *Psychometrika* 1985;50:159-179.
- Chakravarthy SV, Ghosh J. Scale-based clustering using the radial basis function network. *IEEE Trans Neural Netw* 1996;7:1250-1261.
- Frattale Mascioli FM, Rizzi A, Panella M, Martinelli G. Scale-based approach to hierarchical fuzzy clustering. *Sig Proc* 2000;80:1001-1016.
- Kothari R, Pitts D. On finding the number of clusters. *Pattern Recog Lett* 1999;20:405-416.
- Conrad MP. A rapid nonparametric clustering scheme for flow cytometry data. *Pattern Recog* 1987;20:229-235.
- Fu L, Yang M, Braylan R, Benson N. Real-time adaptive clustering of flow cytometric data. *Pattern Recog* 1993;26:365-373.
- Kohonen T. *The self-organising map*. *Proc IEEE* 1990;78:1464-1480.
- Boddy L, Morris CW. Artificial neural networks for pattern recognition. In: Fielding AH, editor. *Machine learning methods for ecological applications*. Boston: Kluwer; 1999. p 37-87.
- Boddy L, Wilkins MF, Morris CW. Pattern recognition in flow cytometry. *Cytometry* 2001. Forthcoming.

#### APPENDIX 1: DKLL ALGORITHM

The DKLL algorithm described by Demers et al. (12) can be summarized as follows:

- For each cluster  $t$ , initialize  $\mu_t$  to a randomly selected data pattern and initialize  $\hat{S}_t$  to the identity matrix.
- Assign each data pattern  $x_i$  to closest cluster  $t$ , i.e., for which  $d_{it} = \min_k d_{ik}$ .
- For each cluster  $t$ , update  $\mu_t$  to the mean of the set of patterns assigned to cluster  $t$ .
- For each cluster  $t$ , calculate the critical distance, i.e., the Euclidean distance to the closest neighboring cluster center:  $c_t = \min_{k \neq t} \|\mu_k - \mu_t\|$ .
- For each cluster  $t$ , update  $\hat{S}_t$  to the covariance matrix of the set of patterns assigned to cluster  $t$  and falling within a distance  $c_t z$  of the cluster center.
- Calculate the objective criterion  $F = \sum_i d_i^2$ , where  $d_i$  is the generalized distance between pattern  $i$  and the centroid of the cluster to which it has been assigned.
- Repeat steps 2-6 until  $F$  attains a minimum value.

The quantity  $z$  is initially small (around 0.25); this is designed to exclude patterns for which the cluster identity is uncertain from influencing the calculation of the scatter matrices. As the algorithm progresses, the cluster assignments are supposed to become more reliable and  $z$  is increased accordingly. In the implementation used here,  $z$  is increased by 0.005 at each iteration, up to a maximum value of 0.5. Any cluster with less than  $3p$  data patterns assigned to it after step 2 (where  $p$  is the data dimensionality) is deemed to be irrelevant and is deleted.

#### APPENDIX 2: AD, ML, MTV, AND SAND ALGORITHMS

The four algorithms proposed by Rousseeuw et al. (16) are variant forms of one generic fuzzy clustering algorithm, summarized below. In this algorithm, each of the  $k$  clusters is characterized completely by the position of its center,  $\mu_t$ , and a scatter matrix,  $\hat{S}_t$ . The distance  $d_{it}$  between the  $i$ -th data pattern  $x_i$  and the cluster center is defined by the Mahalanobis generalized distance metric  $d_{it}^2 = (x_i - \mu_t)^T (\hat{S}_t)^{-1} (x_i - \mu_t)$ . Each data pattern is associated to some extent with all the clusters, not just the closest; the extent of this association is given by  $u_{it}$ , the fuzzy cluster membership of the  $i$ -th data pattern to cluster  $t$ . The cluster memberships range from 0 (no association) to 1 (complete association) and are subject to the constraint that  $\sum_i u_{it} = 1$ ; in other words, for any data pattern, the sum of its cluster memberships over all clusters is 1. Constraining  $u_{it}$  strictly to the values 0 and 1 produces a "crisp" clustering, for which each data pattern is associated exclusively with only one cluster. Allowing

$u_{it}$  to take intermediate values between 0 and 1 results in a fuzzy clustering.

The generic algorithm described by Rousseeuw et al. (16) can be summarized as follows:

1. For each cluster  $t$ , initialize  $\mu_t$  to a randomly selected data pattern and initialize  $\hat{S}_t$  to the identity matrix.
2. For each cluster  $t$ , calculate coefficients  $A_t$  and  $B_t$  (as below).
3. Calculate memberships for each data pattern  $\mathbf{x}_i$ :
  - i. Initialize  $T_i$  to an empty set.
  - ii. For each cluster  $t$ , calculate  $B_{it} \leftarrow B_t d_{it}^2$ .
  - iii. For each cluster  $t$ , calculate

$$u_{it} \leftarrow \frac{1/B_{it}}{\sum_{r \in T_i} 1/B_{ir}} - \frac{1}{B_{it}} \left[ \frac{\sum_{r \in T_i} A_r/B_{ir}}{\sum_{r \in T_i} 1/B_{ir}} - A_t \right], \quad t \notin T_i$$

$$u_{it} \leftarrow 0, \quad t \in T_i$$

- iv. If any  $u_{it} < 0$ , add  $t$  to set  $T_i$  and repeat step (iii)
4. For each cluster  $t$ , update  $\mu_t$  and  $\hat{S}_t$  as follows:

$$\mu_t \leftarrow \frac{\sum_i u_{it}^2 \mathbf{x}_i}{\sum_i u_{it}^2} \quad \hat{S}_t \leftarrow \frac{\sum_i u_{it}^2 (\mathbf{x}_i - \mu_t)(\mathbf{x}_i - \mu_t)^T}{\sum_i u_{it}}$$

5. If no membership  $u_{it}$  changed by more than a small value  $\epsilon$  during the previous iteration, stop; otherwise return to step 2.

The different variants are obtained by changing the manner of calculating the quantities  $A_t$  and  $B_t$ . For the algorithms AD, MTV, and SAND:

$$A_t = \frac{1}{2} \frac{\tau}{\beta} n_t^{\beta-\tau-1} (\theta_t |\hat{S}_t|)^\beta, \quad B_t = n_t^{\beta-\tau-1} (\theta_t |\hat{S}_t|)^\beta$$

where  $p$  is the data dimensionality,  $n_t = \sum_i u_{it}$ , and the parameters  $\beta$  and  $\tau$  are given by  $\beta = 1/p, \tau = 0$  (AD),  $\beta = 1/2, \tau = p/2$  (MTV), and  $\beta = 1/p, \tau = 1$  (SAND). For these three variants, a reasonable a priori assumption is that all clusters should start with the same volume, i.e.,  $\theta_t = 1$  for all  $t$ .

For the ML algorithm, which minimizes a different type of objective function,

$$A_t = -\frac{1}{2} \log |\hat{S}_t|, \quad B_t = 1.$$

For all four algorithms, the value of  $\epsilon$ , indicating convergence, was taken to be 0.01.