

Some Model-Based and Distance-Based Clustering Methods  
for Characterization of Regional Ecological Stressor-Response Patterns  
and Regional Environmental Quality Trends

David B. Farrar

Dissertation submitted to the faculty of the Virginia Polytechnic Institute and State  
University in partial fulfillment of the requirements for the degree of

Doctor of Philosophy  
In  
Statistics

Committee:  
Eric Smith (Chair)  
Ina Hoeschele  
Samantha Prins  
Dan Spitzner  
Keying Ye

August 15, 2006  
Blacksburg, Virginia

Keywords: ecological stressor, ecoregion, environmental  
monitoring, cluster analysis, label-switching

# Some Model-Based and Distance-Based Clustering Methods for Characterization of Regional Ecological Stressor-Response Patterns and Regional Environmental Quality Trends

David B. Farrar

## ABSTRACT

We develop statistical methods for evaluation of regional variation of ecological stressor-response relationships, and regional variation in temporal profiles of water quality, for application to data from monitoring stations on bodies of water. To evaluate regional variation in regression relationships, we use model-based clustering procedures with class-specific regression models. Units for clustering are taken to be basins, or combinations of basins and ecoregions. We rely on a Bayesian formulation and sample the posterior distribution using a Markov chain Monte Carlo algorithm. Two general approaches to the label-switching problem are considered, each leading to procedures that we apply in data analyses. Two applications are presented. We explore some relationships among priors with a Dirichlet distribution for class probabilities. We compare two rank-based criteria for grouping stations according to similarities in temporal profiles. The two criteria are illustrated in a hierarchical cluster analysis based on measurements of a water quality variable.

## Acknowledgements

Funding for my research was received from the USEPA via a Science to Achieve Results (STAR) Grant RD 83136801-0, with principal investigators including Eric Smith and Samantha Prins. The work presented has not been reviewed by the USEPA and so no USEPA endorsement should be inferred. I gratefully acknowledge John Morgan, George Terrell, and the members of my committee, for useful discussions. Jason Hill, Gene Yagow, and Carl Zipper assisted in accessing and interpreting environmental data. I thank my wife for her love and companionship during my graduate studies, and promise projects in the future that are fun for both of us.

# Contents

<b>List of Figures.....</b>	<b>vii</b>
<b>List of Tables .....</b>	<b>ix</b>
<b>Introduction.....</b>	<b>1</b>
<b>Chapter 1 Background on Statistical Modeling and Simulation Procedures.....</b>	<b>6</b>
1.1 Modeling and Inference for Classifications .....	6
1.1.1 Terminology: Clusters, Partitions, Classes, and Classifications .....	6
1.1.2 Sampling Models and Likelihoods.....	8
1.1.3 Some Priors.....	10
1.2 Simulation Methods with Emphasis on Bayes Posterior Sampling .....	12
1.2.1 Bayes Posterior Sampling using MCMC.....	13
1.2.2 Metropolis Search of a Model Space, Eliminating Nuisance Parameters....	15
1.2.3 Partition-Space Random Walks.....	17
1.2.4 Gibbs and other Block-at-a-Time Strategies with Latent Information .....	18
1.2.5 Non-Bayesian Simulation in Cluster Analysis .....	20
1.3 Class-Specific Models.....	20
1.3.1 Multivariate Normal and <i>t</i> Distributions .....	20
1.3.2 Linear and Nonlinear Regression Models .....	21
1.3.3 Semiparametric Regression Models with a Single Predictor .....	21
1.3.4 Semiparametric Regression with Multiple Predictors.....	24
1.4 The Label-Switching Problem and Two Types of Solutions .....	25
<b>Chapter 2 Solutions to the Label-Switching Problem using Relabeling and Label-Invariant Summarization.....</b>	<b>28</b>
2.1 Partitions and Classifications, with Matrix Representations.....	29
2.2 A Label-Invariant approach to post-processing .....	30
2.2.1 The ECCP matrix.....	30
2.2.2 Evaluating the ECCP Using Clustering and Ordination.....	31
2.2.3 Label-Invariant Identification of Clusters from Simulation Output, with Evaluation of Classification Uncertainty.....	32
2.2.4 Conditional Estimation of Class-Specific Parameters .....	32
2.3 Relabeling and Related Computations .....	33
2.3.1 The Alignment Matrix .....	33
2.3.2 Matrix Representation of a Criterion for Aligning Two Classifications .....	34
2.3.3 A Criterion for Alignment of a Sample of Classifications .....	34

2.3.4	<i>A Relabeling Algorithm</i> .....	35
2.5	Discussion .....	37
2A	Appendix: Technical Results Related to the Ordination Approach .....	38
<b>Chapter 3</b>	<b>Ecological Applications of Model-Based Clustering with Class-Specific Linear and Semi-Parametric Regressions</b> .....	<b>40</b>
3.1.	Bayes Model and Posterior Sampling Procedure .....	42
3.1.1	<i>Sampling Model and Prior</i> .....	42
3.1.2	<i>Posterior Sampling Procedure</i> .....	43
3.1.3	<i>Summarizing the Posterior Sample</i> .....	46
3.2	An Application to Data from Maryland with Class Spline Regressions.....	48
3.2.1	<i>Data</i> .....	48
3.2.2	<i>Preliminary Analysis, with Comparison of Physiographic Regions</i> .....	52
3.2.3	<i>Model-Based Clustering with Additive B-Splines</i> .....	60
3.2.4	<i>Results</i> .....	61
3.2.5	<i>Discussion</i> .....	62
3.3	An Application to Data from Ohio, with Class Linear Models.....	79
3.3.1	<i>Data</i> .....	79
3.3.2	<i>Model-Based Clustering with Linear Class Models</i> .....	80
3.3.3	<i>Results</i> .....	81
3.3.4	<i>Discussion</i> .....	82
3.4	Discussion of the Model-Based Clustering Approach, with an Analysis Allowing Class-Specific Variances.....	94
3A.1	Appendix: Unit Codes and Numbers of Stations for Two Datasets.....	98
<b>Chapter 4</b>	<b>Some Comparisons among Priors with Dirichlet-Distributed Class Probabilities</b> .....	<b>99</b>
4.1	Product-Multinomial Priors and Priors with Dirichlet Distributed Class probabilities.....	100
4.2	Partition-Space Random Walk Algorithm for Model with Dirichlet Class Probabilities.....	102
4.3	A Preliminary Monte Carlo Study of Sampling Properties.....	103
4.3.1	<i>Procedure</i> .....	104
4.3.2	<i>Results and Discussion</i> .....	106
4.4	Empirical Comparisons Based on the Maryland and Ohio Datasets.....	109
4.5	Implications for Relative Cluster Sizes .....	112
4A.1	Appendix: Some Properties of the Dirichlet distribution.....	113
4A.2	Appendix: Technical Remarks on Priors for a Classification .....	114

<b>Chapter 5</b>	<b>Clustering Monitoring Stations Based on Two Rank-Based Criteria of Similarity of Temporal Profiles .....</b>	<b>117</b>
5.1	Cluster Analysis Methodology .....	119
5.1.1	<i>Hierarchical Clustering of Monitoring Stations Based on Similarities in Temporal Profiles .....</i>	<i>119</i>
5.1.2	<i>A Criterion of Trend Similarity Related to a Standard Trend Homogeneity Test.....</i>	<i>120</i>
5.1.3	<i>Clustering with Ward's Method and Transformed Plotting Heights.....</i>	<i>122</i>
5.1.4	<i>An Approach Based on Concordant and Discordant Changes over Time .</i>	<i>123</i>
5.2	An Empirical Comparison of the Two Criteria .....	125
5.2.1	<i>Data on Dissolved Oxygen in the James River of Virginia .....</i>	<i>125</i>
5.2.2	<i>Results.....</i>	<i>126</i>
5.3	Discussion .....	128
5A	Appendix: Station codes and geographic coordinates for 71 stations used in the analysis. ....	146
<b>Chapter 6</b>	<b>Discussion and Extensions.....</b>	<b>148</b>
<b>References.....</b>		<b>150</b>
<b>Appendix I.</b>	<b>R Functions Used in Model-Based Clustering.....</b>	<b>161</b>
<b>Appendix II.</b>	<b>R Functions Used in Clustering of Temporal Profiles.....</b>	<b>195</b>

## List of Figures

Figure 3-1	<i>Locations of Maryland monitoring stations and clustering units.</i>	50
Figure 3-2	<i>Distributions for two predictor variables, by physiographic region.</i>	51
Figure 3-3	<i>2-dimensional smooth relating BIBI to two land use variables based on combined data.</i>	54
Figure 3-4	<i>3-Dimensional scatterplot relating BIBI to two land use predictors.</i>	55
Figure 3-5	<i>Urbanization component from an additive model fitted to the combined data.</i>	56
Figure 3-6	<i>Agricultural component from an additive model fitted to the combined data.</i>	57
Figure 3-7	<i>Relationship of BIBI to urbanization for two physiographic regions.</i>	58
Figure 3-8	<i>Relationship of BIBI to agricultural usage for two physiographic regions.</i>	59
Figure 3-9	<i>Ten Metropolis chains each for 2- and 3-class models.</i>	65
Figure 3-10	<i>Ten Metropolis chains each, for 4- and 5-class models.</i>	66
Figure 3-11	<i>Convergence diagnostic plot for 5 selected chains from 3-class solution.</i>	67
Figure 3-12	<i>Measures of model quality related to the number of classes assumed.</i>	68
Figure 3-13	<i>Dendrogram and ordination plot based on ECCP from 2-class model.</i>	69
Figure 3-14	<i>Dendrogram and ordination plot for 3-Class model.</i>	70
Figure 3-15	<i>Dendrogram and ordination plot based on 5 selected chains, for 3-class model.</i>	71
Figure 3-16	<i>Dendrogram and ordination plot for 4-class model.</i>	72
Figure 3-17	<i>Relationship of BIBI to urbanization, based on 2-class model.</i>	73
Figure 3-18	<i>Relationship of BIBI to agricultural use, based on 2-class model.</i>	74
Figure 3-19	<i>Relationship of BIBI to urbanization, based on 3-class model.</i>	75
Figure 3-20	<i>Relationship of BIBI to agricultural use, based on 3-class model.</i>	76
Figure 3-21	<i>Relationship of BIBI to urbanization, based on 5 selected chains from 3-class results.</i>	77
Figure 3-22	<i>Relationship of BIBI to agricultural use, based on 5 selected chains from 3-class results.</i>	78
Figure 3-23	<i>Locations of Ohio stations and basins.</i>	84
Figure 3-24	<i>Metropolis chains for models with 2-4 classes.</i>	85
Figure 3-25	<i>Measures of model quality related to the number of classes assumed.</i>	86
Figure 3-26	<i>Dendrogram and ordination plot based on ECCP from 2-class model.</i>	87
Figure 3-27	<i>Dendrogram and ordination plot based on ECCP from 3-class model.</i>	88
Figure 3-28	<i>Dendrogram and ordination plot based on ECCP from 4-class model.</i>	89
Figure 3-29	<i>Estimated posterior distributions of class intercepts from 2-class model.</i>	90
Figure 3-30	<i>Estimated posterior distributions of regression coefficient for dissolved oxygen, for 2-class model.</i>	91
Figure 3-31	<i>Estimated posterior distributions for regression coefficient for pH, for 2-class model.</i>	92
Figure 3-32	<i>Estimated posterior distributions for regression coefficient for zinc, for 2-class model.</i>	93
Figure 3-33	<i>Estimated posterior distributions for regression coefficient for QHEI, for 2-class model.</i>	94

Figure 3-34 <i>Ten Metropolis chains for 3-class model with heterogeneous variances (Maryland dataset).</i> .....	97
Figure 4-1 <i>Bias and mean square error results from simulation.</i> .....	109
Figure 4-2 <i>Log-likelihood by for Maryland dataset.</i> .....	112
Figure 5-1 <i>Locations of monitoring stations for the James River basin.</i> .....	131
Figure 5-2 <i>Detailed map of region containing Pagan River and Jones Creek stations.</i>	132
Figure 5-3 <i>Time series of dissolved oxygen for stations in the Pagan River Basin and along the James River channel.</i> .....	133
Figure 5-4 <i>Dendrogram based on the dissimilarity criterion, Ward's linkage criterion, and the conventional scaling of branch heights.</i> .....	134
Figure 5-5 <i>Dendrogram with branch heights re-scaled.</i> .....	135
Figure 5-6 <i>Average linkage dendrogram based on RTepsi.</i> . .....	136
Figure 5-7 <i>Scatterplot comparing two distances for 71 stations.</i> .....	137
Figure 5-8 <i>Dendrograms for Pagan River and Jones Creek stations, based on the two criteria.</i> .....	138
Figure 5-9 <i>Comparison of profiles for the two stations.</i> .....	139
Figure 5-10 <i>Temporal profiles for individual stations in four clusters identified by clustering based on the ZDiff<sup>2</sup> criterion.</i> .....	140
Figure 5-11 <i>Temporal profiles for individual stations in four clusters identified by clustering based on the RTepsi criterion.</i> .....	141
Figure 5-12 <i>Coordinates of stations in 4 clusters identified by clustering based on the ZDiff<sup>2</sup> criterion.</i> .....	142
Figure 5-13 <i>Coordinates of stations in 4 clusters identified by clustering based on the RTepsi criterion.</i> .....	143
Figure 5-14 <i>Hypothetical temporal profiles illustrating differences between the two criteria.</i> .....	144
Figure 5-15 <i>Relation of average RTepsi to number of clusters.</i> .....	145



## List of Tables

Table 1-1 <i>Notation for classifications.</i> .....	8
Table 2-1 <i>Alignment matrix for hypothetical example.</i> .....	33
Table 3-1 <i>Estimated probabilities of class membership for each unit in the Maryland data, based on selected chains from 3-class model.</i> .....	64
Table 3-2 <i>Classification of Ohio basins based on 2-class model.</i> .....	83
Table 3A-1 <i>Unit codes and counts of stations in the two datasets.</i> .....	98
Table 4-1 <i>Monte Carlo assessment of sampling error with Dirichlet distributed class probabilities.</i> .....	108
Table 4-2 <i>Sensitivity of clustering results to Dirichlet parameter, based on two datasets.</i> .....	111
Table 5-1 <i>Homogeneity of trend tests with pooling over clusters.</i> .....	128
Table 5A-1 <i>James River monitoring stations.</i> .....	147

## Introduction

We develop statistical methods for evaluation of regional variation of ecological stressor-response relationships, and regional variation in temporal profiles of water quality. Data for such analyses will represent monitoring stations located on bodies of water. For each station, measurements may be available for chemical concentrations, hydrological variables, frequencies of biological taxa, and other variables. Some variables, such as pollutant concentrations, may be termed “stressors” because of a possible relationship to adverse ecological effects.

Regression models may be used to relate stressors and other ecological predictor variables to measures of ecological quality. We incorporate regional specificity, by use of model-based clustering methods that assume an unknown classification of monitoring stations, with class-specific regressions. Formally, the data on ecological quality for  $n$  stations may be represented by observations  $y_1, \dots, y_n$ . Each of these is assumed to have been drawn from one of  $k$  classes -- we do not know which. Conditioning on the event that the  $i$ th station falls in the  $j$ th class, the observation  $y_i$  is a realization of a random variable with density  $f(\cdot; \xi_j, \eta, \mathbf{x}_j)$  where  $\xi_j$  is an unknown class-specific parameter,  $\eta$  is a parameter in common, and  $\mathbf{x}_j$  is a fixed, known vector of measurements on predictor variables. The expected response in a class may be a linear or nonlinear function of  $\mathbf{x}_j$ .

Procedures allowing class-specific regressions have been used in various fields (*e.g.*, DeSarbo and Cron, 1988; Viele and Tong, 2002; Qin and Self, 2006), and have been used to evaluate regional ecological patterns in particular (Lamon and Stow, 2004; Lipkovich, 2002). In the context of ecological data, several justifications can be given:

- Detection of apparent regional patterns may stimulate and focus research by suggesting scientific hypotheses related to regional effects of environmental

- variables. Such hypotheses may be evaluated based on more refined analysis of the data, or by generation of additional data.
- Region-specific effects of a stressor may be overlooked if regional variation is ignored, so that we conclude incorrectly that a variable is unimportant.
  - An apparent pattern, based on an analysis that ignores regional variation, may actually result from poorly understood regional effects. For example, an apparent deviation from monotonicity of a stressor-response relationship might actually be due to a level of the response variable, in some region, not consistent with a global, monotone relationship.
  - Some support for region-specific regressions is provided by the suggestion that ecoregions should be defined so as to be homogeneous with respect to ecological processes (McMahon *et al.*, 2001).
  - Considering the possibilities for interactions and nonlinearities, it is possible that a useful model will assume that a simple, conventional model such as linear regression holds regionally rather than globally.

An indication of the importance given to regional variation is the development, by government agencies, of standard ecoregions (Bailey, 1996; McMahon *et al.* 2001; Omernik, 1987; Omernik and Bailey, 1997; USEPA, 2003; Woods *et al.*, 1999). Ecoregions have been used in various ways in environmental analysis (*e.g.*, Larsen *et al.*, 1988; Morgan and Cushman, 2005; Norton, 1999). In particular, groups of monitoring stations identified using cluster analysis results are often compared to standard ecoregions (*e.g.*, Poff and Allan, 1995).

Several approaches are encountered in literature on model-based clustering, for estimating the number of classes ( $k$ ) or accounting for uncertainty in  $k$  (Fraley and Raftery, 1998; Richardson and Green, 1997; Stephens, 2000; Ter Braak *et al.*, 2003). We currently use models with  $k$  fixed, and compare results obtained with different values of  $k$ . It seems that we obtain meaningful results with  $k$  fixed at values appropriate for regional variation on approximately a spatial scale suggested by McMahon *et al.* (2001).

In addition to clustering according to similarities in regression relationships, we explore clustering according to similarities in temporal profiles. To provide for evaluation of water quality trends, a water quality variable may be measured at a series of points over time. Multivariate methods such as cluster analysis may be useful for summarization of such data and for detection of spatiotemporal patterns. We propose hierarchical clustering methods that group stations according to similarities among temporal profiles, combining standard clustering algorithms with two proposed, rank-based criteria of similarity.

Following are some important aspects of particular chapters. (For various topics cited, literature references are found in the chapters indicated.) Chapter 1 provides background on relevant statistical models and simulation procedures. Semiparametric regression models are reviewed, particularly for use in ecological stressor-response modeling. Bayesian posterior sampling is reviewed at a basic level, emphasizing Markov chain Monte Carlo (MCMC).

Chapter 2 is a study of solutions to the label-switching problem. That term suggests the event that the same classification recurs in the posterior sample, with classes indexed differently for different occurrences. A remedy suggested by some authors is *relabeling*, or adjustment of class indices for each classification in the sample. We propose a relabeling algorithm that maximizes a simple measure of agreement among classifications. In addition, we develop methods for visualizing the information in a sample of classifications, which do not depend on how clusters are labeled. Our procedures for relabeling and label-invariant visualization, developed in Chapter 2, are found to be useful in applications presented in Chapter 3. In addition to application to Bayes posterior sampling, we observe that the methods of Chapter 2 apply to simulation procedures with various conceptual foundations, which generate random samples of classifications, for evaluation of the stability, sampling error, or uncertainty of cluster analysis results.

In Chapter 3 we apply model-based clustering to two sets of data collected by monitoring streams and rivers in the states of Ohio and Maryland (USA). For each application the response is an index of ecological quality developed for use by a government agency for evaluation of ecological quality.

For the Maryland dataset, preliminary analysis suggested a nonlinear relationship of the response to the predictors. Therefore our model with class linear regressions is extended by use of *B*-splines for specific predictors, treated as additive across predictors. This extension represents the first model-based cluster analysis known to us with multiple predictors and a semiparametric representation of each predictor.

Taken together, the two applications suggest that our approach may discriminate between predominantly lowland streams and predominantly upland streams, with lowland streams tending to have lower ecological quality as measured. For example, a cluster identified from the Ohio data has a high fraction of stations from lowlands adjacent to Lake Erie. A tendency to separate lowland streams may account also for a cluster based on the Maryland data, which includes a low-lying basin in the western Piedmont, along with stations from the coastal plain. Our analysis of the Maryland data provides a possible example where a pattern apparent in an analysis ignoring regional variation may be attributable to a subset of the data.

To promote an initial focus on ecological applications, the priors used in our applications have been chosen largely for convenience. For the unknown classification, assuming that each classification is equally probable leads to relatively simple computations. In Chapter 4 we embed such a prior in a more general family with a Dirichlet hyperprior for class probabilities. Relationships among priors in the family are explored conceptually, based on limited comparisons employing the Maryland and Ohio datasets, and with a preliminary simulation study comparing sampling properties.

Chapter 5 presents our procedures for clustering stations based on similarities in temporal profiles. We study two rank-based criteria of similarity, which we use in hierarchical

clustering of monitoring stations. A criterion derived from a standard test for trend heterogeneity is viewed as complementary to standard environmental trend evaluation. However, the standard criterion may not be sensitive to certain patterns that may have practical significance, for example if the sign of a trend reverses at a particular point in time in a particular region, as may happen if a local environmental problem is remediated. Therefore we introduce a second criterion of similarity, expected to be sensitive to more diverse patterns, based on concordance of changes between pairs of stations, for pairs of measurement times. The two criteria are illustrated and compared based on application to measurements of dissolved oxygen in the James River of Virginia, USA. While our results for the two methods are not very similar overall, the methods agree in identifying a particular locality with pronounced negative trends evident for multiple stations.

## Chapter 1

### Background on Statistical Modeling and Simulation Procedures

We provide background on modeling with classifications and class-specific regressions. First, we first introduce some concepts and terminology. Some sampling models and Bayesian models are introduced. Semiparametric regression procedures are reviewed from the viewpoint of modeling nonlinear stressor-response relationships, particularly focusing on approaches amenable to incorporation in model-based clustering. We briefly discuss some options for semiparametric modeling with multiple predictors. Bayesian posterior sampling is reviewed at a basic level, and we also mention some non-Bayesian simulation approaches. Procedures are reviewed that are useful for summarizing a Bayes posterior sample, particularly procedures that take the so-called label-switching problem into account.

#### 1.1 MODELING AND INFERENCE FOR CLASSIFICATIONS

##### 1.1.1 Terminology: Clusters, Partitions, Classes, and Classifications

We study cluster analysis methods that classify  $n$  units – in our context, monitoring stations or groups of monitoring stations -- into  $k$  disjoint sets. Groups of units can be termed *clusters*, and the clusters define a *partition*. We distinguish between a partition and a *classification* or *labeled partition*. A partition is given by stating, for each pair of units, whether the members of the pair are classified together or separately. The specification does not require labeling of clusters. A classification, in our terminology, is obtained from a partition by assigning a unique label or index to each cluster. For any partition comprising  $k$  clusters, there are  $k!$  classifications associated with permutations of  $k$  given labels. In our context, the “labels” will actually be  $1, \dots, k$ , so that labeling is effectively ordering or indexing. Clusters are inevitably ordered in computer algorithms, according to array indices or other memory locations. In any case, references to labels are conventional.

While the term “cluster” is associated with groups of units, the term *class* will be associated with distinct populations posited in a model, as in the expression “*k*-class model.” To understand the distinction, consider the quantity “probability that Unit *i* belonging to Class *j*.” In our applications, the quantity is estimated by the fraction of sample from a posterior distribution, with Unit *i* assigned to Class *j*. To compute the quantity, different clusters in the sample may be associated with the same class, and a given cluster may be associated with different classes. However, in many situations the terms “class” and “cluster” can be interchanged without causing confusion.

We now give three representations of a classification, each with advantages in particular situations. We illustrate each based on the same hypothetical case with 4 units and 2 clusters, with Units 1 and 2 assigned to Class 1 and Units 3 and 4 to Class 2.

- A classification may be represented as an *n*-tuple  $\gamma = (\gamma_1, \dots, \gamma_n)$  giving a class index for each unit, for example  $\gamma = (1, 1, 2, 2)$ .
- A classification may be represented using indicator variables

$$z_{ij} = \begin{cases} 1 & \text{if } \gamma_i = j, \\ 0 & \text{otherwise.} \end{cases}$$

- A classification may be represented as a *k*-tuple  $(C_1, \dots, C_k)$ , where  $C_j$  is a cluster, for example  $(\{1, 2\}, \{3, 4\})$ .

The first representation  $\gamma$  seems particularly natural, so we will use it for our default representation.

Table 1.1 summarizes some notation. (Additional, matrix representations of partitions and classifications will be presented in Chapter 2.)



**Table 1-1** *Notation for classifications.*

<b>Symbol</b>	<b>Definition</b>
$N$	Number of units to be classified.
$K$	Number of classes in a model.
$C_j \subset \{1, \dots, k\}$	$j$ th cluster of a classification.
$n_j$	Size of $C_j$ (count of units).
$z_{ij} \in \{0, 1\}$	Cluster membership indicator, equal to 1 if $i \in C_j$ , else equal to 0.
$\gamma$	Classification as an $n$ -tuple of class indices. Also, our generic symbol for a classification.

### 1.1.2 *Sampling Models and Likelihoods*

For the statistical models we use, an observation is treated as a realization of a class-specific distribution, where the classification  $\gamma$  is treated as unknown. In this section we provide background on important variants of this idea, associated with different sampling models and different likelihood functions. Although we currently emphasize Bayesian posterior sampling, we briefly describe common maximum-likelihood procedures.

In general, observations will be assumed drawn from some parameterized distribution family with unknown class-specific parameter  $\xi_j$ , and possibly a parameter  $\eta$  equal in value in each class. In our applications, class-specific parameters include regression coefficients. Possible examples of a common parameter include a common residual variance in a regression model, or some regression coefficient assumed equal in value among classes.

According to the approach we have adopted for our applications,  $\gamma$  is viewed as fixed in the sampling model. (It may still be random in a Bayesian approach, if it is assigned a prior.) Given  $\gamma$ , the observations are viewed as conditionally independent. The likelihood based on observing  $y_1, \dots, y_n$  is then written

$$L(\xi_1, \dots, \xi_k, \eta, \gamma; y_1, \dots, y_n) = \prod_{j=1}^k \prod_{i \in C_j} f(y_i; \xi_j, \eta) \quad (1.1)$$

While (1.1) is taken to be the likelihood in our applications, an alternative that is more common in current model-based clustering uses a *finite mixture* model. As we would define that model, it assumes that data have been generated in two random stages: The first stage selects one of  $k$  subpopulations, often termed *components*. We let  $\tau_j$  denote the probability of choosing class  $j$  in the first step ( $j = 1, \dots, k$ ). (The parameters  $\tau_1, \dots, \tau_k$  are often termed *mixing fractions*.) In the second stage the observations are generated by sampling component-specific distributions. The likelihood is then

$$L(\xi_1, \dots, \xi_k, \eta, \tau_1, \dots, \tau_k; y_1, \dots, y_n) = \prod_{i=1}^n \sum_{j=1}^k \tau_j f(y_i; \xi_j, \eta). \quad (1.2)$$

However, there is a possibility for confusion about the meaning of “finite mixture model.” Some Bayesian posterior sampling methods can be used whether the mixing fractions are interpreted as relative frequencies, or as parameters of a prior, despite the difference in likelihood. It seems that in some applications the intended interpretation of the mixing fractions is actually unclear. When we refer to the general literature of methods billed as “finite mixture” we will enclose the term in quotation marks. Specific situations such as development of a Jeffreys prior may require the use of (1.2).

Special maximum-likelihood procedures are associated with (1.1) or (1.2). For given data, each can be maximized using an appropriate version of expectation-maximization (EM) algorithm. Optimization of (1.2) is a standard application of EM (Dempster *et al.*, 1977). In the maximum-likelihood context (1.1) is sometimes termed *classification likelihood*. Optimization can be accomplished using an EM version termed *classification EM* (CEM), which alternates between estimating distribution parameters with the classification fixed, and estimating the classification with distribution parameters fixed (Govaert and Nadif, 1996; cf. Marriott, 1975; Bryant and Williamson, 1978).

Whether we adopt (1.1) or (1.2) as the likelihood, the model may not be identified without additional information such as an order constraint on class parameters, or a prior that differs for class-specific parameters. The problem has been termed *label-switching* (Celeux *et al.* 2000; Stephens, 2000). This term reflects the possibility that with an

MCMC implementation the same actual partition recurs in the posterior sample, with clusters labeled differently for different occurrences. We will discuss some remedies for this problem.

### 1.1.3 *Some Priors*

We implement joint Bayesian inference for an unknown classification, and for parameters of class-specific regressions. The approach requires priors for all unknowns.

An obvious prior for a classification  $\gamma$  would assign equal probability to each allowed classification. This is the approach taken for our ecological applications in Chapter 3. An alternative explored in Chapter 4 introduces class probabilities  $\tau_1, \dots, \tau_k$  satisfying  $\sum_{j=1}^k \tau_j = 1$  and  $\tau_j \geq 0$  ( $j = 1, \dots, k$ ), which may have a Dirichlet distribution.

For our applications we have adopted flat priors for class regression parameters. Literature on finite mixture models suggests some emphasis on automated selection of proper conjugate priors (Roeder and Wasserman, 1997). In particular, if Bayes factors are to be used in comparing models with different dimensions, priors should be proper for parameters common to the models compared. Wasserman (2000) has suggested that conventional “uninformative” priors, including flat priors and Jeffreys priors, are useful for obtaining asymptotic confidence intervals in the finite mixture context.

For Bayesian applications with class-specific priors that are improper -- do not integrate to 1 over the parameter space -- it is desirable to demonstrate propriety of the posterior.

For unknown model parameter  $\theta$  and data  $\mathbf{D}$  suppose that

$$\int_{\theta} \pi(\theta) f(\mathbf{D} | \theta) d\theta < \infty$$

where  $\pi(\theta)$  denotes the density for the prior (possibly improper) and  $f(\mathbf{D} | \theta)$  the conditional density for the data. Then  $\pi(\theta) f(\mathbf{D} | \theta) / \int_{\theta} \pi(\theta) f(\mathbf{D} | \theta) d\theta$  may be treated as the posterior density for  $\theta$  (Robert, 2001, p. 28).

Posterior propriety can be studied for mixture models by expressing the mixture likelihood as a sum over classifications, then integrating (Roeder and Wasserman, 1997, Wasserman, 2000). The same approach applies to more general models with unknown classifications. This leads to a requirement of a minimum cluster size, which is easily imposed when relying on posterior sampling.

The following simple example from Roeder and Wasserman, involving a location-scale mixture with two classes, illustrates an approach to propriety. The  $j$ th class-conditional distribution is  $N(\mu_j, \sigma_j^2)$  ( $j = 1, 2$ ). Let  $\xi_j = (\mu_j, \log \sigma_j)$  and let  $\gamma$  denote a single classification in a set  $H$  of allowed classifications. The mixture likelihood can be written

$$L(\xi_1, \xi_2, \tau_1) = \sum_{\gamma \in H} f(y | \xi_1, \xi_2, \gamma) f(\gamma | \tau_1).$$

For model identification, RW impose an order constraint on the class means, say  $\mu_1 < \mu_2$ . The standard non-informative prior has density  $\pi(\xi_1, \xi_2) = I(\mu_1 < \mu_2)$ . The propriety condition is

$$\begin{aligned} \infty &> \int_{\tau_1, \xi_1, \xi_2} L(\xi_1, \xi_2, \tau_1) \pi(\tau_1) \pi(\xi_1, \xi_2) d\tau_1 d\xi_1 d\xi_2 \\ &= \int_{\tau_1} \left[ \sum_{\gamma \in H} \left[ \int_{\xi_1, \xi_2} f(y | \xi_1, \xi_2, \gamma) d\xi_1 d\xi_2 \right] f(\gamma | \tau_1) \right] \pi(\tau_1) d\tau_1. \end{aligned}$$

Thus for a classification  $\gamma$  we require

$$\int_{\xi_1, \xi_2} f(y | \xi_1, \xi_2, \gamma) d\xi_1 d\xi_2 < \infty.$$

Roeder and Wasserman remark on the case of a classification where all units belong to the same cluster. Then the integral has a component, for the “empty” cluster, which is simply the integral of the prior, and which will be infinite for any empty clusters. Roeder and Wasserman remark that then “the data provide no information about  $\mu_1$  or  $\sigma_1$ ” (where they take the empty cluster to be cluster 1).

More generally, with the classification fixed the integral should be finite. Our class models are linear models. For propriety with the flat priors we use, it suffices that there are positive degrees of freedom for estimation of variances, and non-singular class design matrices (O’Hagan, 1994).

A minimum cluster size is easily implemented when relying on posterior sampling (Diebolt and Robert, 1994; Lipkovich, 2002; Wasserman, 2002). The algorithm is modified to reject classifications that do not satisfy the minimum cluster size. A classification EM algorithms can be modified by selecting on each iteration an update that satisfies the minimum cluster size. The appropriate modification of a mixture EM algorithm is not immediately evident.

In our applications, we require a minimum cluster size larger than required for propriety. We are concerned that cluster-specific design matrices may sometimes be too poorly conditioned, if too few observations are required per cluster, although the posterior may be proper.

## 1.2 SIMULATION METHODS WITH EMPHASIS ON BAYES POSTERIOR SAMPLING

We digress to provide some background on posterior sampling before resuming our discussion of class-specific models. In addition, we mention some frequentist simulation procedures that may be used to evaluate stability or sampling error associated with results of a cluster analysis.

### 1.2.1 Bayes Posterior Sampling using MCMC

Simulation procedures may be used to sample the posterior distribution. Useful overviews include Chib and Greenburg (1995), Gilks *et al.* (1996), Rossi *et al.* (2005), and Tanner (1996). The simulation approach used most is Markov chain Monte Carlo (MCMC), which we now describe at a basic level. The idea is to simulate a Markov chain that has alternative models as its state space, and the posterior distribution of interest as its equilibrium distribution. A sufficiently lengthy realization (a “chain”) can be treated as a sample from the posterior. If changes of the model are not too large from one step to the next, sampling may be focused on models that have substantial likelihood.

An appropriate MCMC technique should have a unique equilibrium distribution, over a space of models, which should equal the posterior distribution in light of data  $\mathbf{D}$ . To establish terminology, consider the case of a finite model space, where the essential ideas are clear (Lange, 1999). Consider selection from among  $n_M$  models denoted  $M_1, \dots, M_{n_M}$ .

A Markov chain is specified by giving transition probabilities  $\{p_{ij}\}$ , where  $p_{ij}$  is the probability that the next model be  $M_j$  when the current model is  $M_i$ , independent of outcomes before the current step. While the process should be ergodic, in practice it may be convenient to check the stronger condition of reversibility, which holds if the process is aperiodic and irreducible. Then posterior probabilities  $\pi(M_1 | \mathbf{D}), \dots,$

$\pi(M_{n_M} | \mathbf{D})$  given model equilibrium frequencies for the Markov chain if they satisfy “detailed balance” equations

$$\pi(M_i | \mathbf{D}) p_{ij} = \pi(M_j | \mathbf{D}) p_{ji}. \quad (1.3)$$

The MCMC algorithms studied here are Metropolis-Hastings algorithms. On a given iteration a “proposed” model is generated, which may be accepted or rejected. The acceptance step is stochastic. If the proposed model is not accepted then the current model is carried forward to the next iteration.

In the simplest case the proposal rule is symmetric:  $q_{ij} = q_{ji}$  where  $q_{ij}$  denotes the probability of proposing model  $M_j$  when the current model is  $M_i$ . The acceptance rule can be expressed in terms of the relative odds (RO) comparing proposed model  $M_p$  to current model  $M_c$  in light of data  $\mathbf{D}$ ,

$$\text{RO}_{pc} = \frac{\pi(M_p / \mathbf{D})}{\pi(M_c / \mathbf{D})}$$

where a value greater than 1 favors  $M_p$ . The proposal is accepted if it has higher posterior probability, *i.e.*, if  $\text{RO}_{pc} > 1$ . Otherwise acceptance is based on a Bernoulli trial with probability of acceptance  $\text{RO}_{pc}$ . This decision scheme is conventionally summarized by writing the acceptance probability as

$$\min \{ \text{RO}_{pc}, 1 \}.$$

The posterior model probabilities are readily seen to satisfy (1.3).

According to current terminology in statistics, the algorithm with symmetric proposals is termed a *Metropolis* algorithm. Asymmetric proposals may be useful in some situations and are easily accommodated by adjustment of the acceptance probability. The general algorithm, with possibly asymmetric proposals, is termed *Metropolis-Hastings* (MH).

We stress the flexibility of MH procedures. Implementation requires only relative probabilities comparing pairs of models. This feature may be valuable particularly for implementing models with constraints on the supports of model unknowns, such as a minimum cluster size. When comparing a pair of models that both satisfy our constraints, the relative odds is with the constraint imposed is equal to the relative odds with the constraint relaxed.

Additional flexibility is provided by the Gibbs sampler, and related block-at-a-time MH variants, discussed below, which can exploit simplifications that result from fixing some model unknowns, for example from fixing the classification, in our models.

### 1.2.2 Metropolis Search of a Model Space, Eliminating Nuisance Parameters

We discuss some methods that may be used to explore a space of partitions, with the parameters of class-specific distributions eliminated. An algorithm of this type was suggested by Lipkovich (2002); however, our discussion will be somewhat more general.

We may define a Metropolis algorithm with states  $M_1, \dots, M_{n_M}$  corresponding to a finite set of models. These may have prior probabilities  $\pi_1, \dots, \pi_{n_M}$ . Model comparisons will usually involve nuisance parameters  $\theta_i \in \Theta_i$ . According to the strict Bayesian approach, a chain may be defined in the space of models, with parameters eliminated by integration. For a given step in the chain, let  $M_c$  denote the current model and  $M_p$  the proposed model. With a symmetric proposal rule, the acceptance probability relies on posterior odds ratio

$$\text{RO}_{pc}(\mathbf{D}) = \frac{\pi_p}{\pi_c} \text{BF}_{pc}(\mathbf{D}).$$

Here the data enter via the Bayes factor

$$\text{BF}_{pc}(\mathbf{D}) = \frac{I(\mathbf{D} | M_p)}{I(\mathbf{D} | M_c)}$$

where  $I(\mathbf{D} | M_i)$  denotes the marginal density of data  $\mathbf{D}$  under model  $M_i$ . In the simple case of no nuisance parameters the Bayes factor is simply a likelihood ratio. Otherwise the nuisance parameter is “integrated out” by we computing

$$I(\mathbf{D} | M_i) = \int_{\Theta_i} f_i(\mathbf{D} | \theta_i) \pi_i(\theta_i) d\theta_i$$

where  $\pi_i(\theta_i)$  denotes the prior for the parameter in class  $i$ . It is important that model comparisons depend on our priors for nuisance parameters.

Important practical issues relate to existence and computation of Bayes factors (Kass, 1993; Kass and Raftery, 1995; Robert, 2001). Bayesian model comparisons may be problematic for models with parameters of different dimension, in case of improper priors for model parameters, even where the posteriors are proper for individual models



(Kass and Raftery, 1995; Roeder and Wasserman, 1997). If the prior density is constant in value, the posterior is defined only up to a constant, so that Bayes factors are undefined.

$I(\mathbf{D} | M_i)$  can be evaluated analytically when the prior  $\pi_i(\theta_i)$  is conjugate to the likelihood  $f_i(\mathbf{D} | \theta_i)$  (references in Kass and Raftery, 1995). More generally, the integral can be approximated using Monte Carlo procedures. A simple approach is to average the likelihood values for a sample from the prior. Lewis and Raftery (1997) consider this approach to be too computationally expensive. Most workers prefer procedures that focus the sampling to some degree in a region of the parameter space with relatively high likelihood. Importance sampling may be considered.

An appealing idea is to compute some kind of average of the likelihood values in a posterior sample, for use in model comparisons. An argument from Kass and Raftery (1995) leads to use of the harmonic mean likelihood value. However, the harmonic mean is considered unstable particularly in case of some small likelihood values. Some remedies are discussed by Kass and Raftery and by Robert (2001).

Asymptotic approximations of  $I(\mathbf{D} | M_i)$  in widespread use include Schwarz and Laplace approximations (Boone, Ye, and Smith, 2005; Kass, 1993; Kass and Raftery, 1995; Robert, 2001; Schwarz, 1978). Kass and Raftery (1995) discuss some variants of the Laplace approximation. In effect, Schwarz and Laplace approximations add a penalty to the optimized log-likelihood or to the logarithm of the optimized posterior density. Laplace approximation uses information, not used by the Schwarz criterion, on the curvature of the likelihood or posterior density in the region of the optimum. An assumption for each approximation is the existence of an optimum in the interior of the parameter space (Tierney and Kadane, 1986).

The Schwarz approximation is the basis of the Bayes Information Criterion (BIC), which is commonly used as a criterion for model selection. In practice the BIC is commonly

used without invoking its Bayesian origin, as a means to penalize a maximized likelihood, to prevent over-fitting. In particular, the BIC is sometimes used to choose the number of classes for a finite mixture model (Fraley and Raftery, 1998, 2002; Ter Braak *et al.*, 2003).

The likelihood penalty associated with the BIC does not depend on a specific prior for the nuisance parameter, except for a dependence on the dimension of the nuisance parameter. Because the specific prior is eliminated, BIC provides some common ground with frequentist practice, where optimized likelihoods -- termed “profile likelihoods” -- are widely used, sometimes with penalties that depend on the number of adjustable parameters, to avoid over-fitting.

### *1.2.3 Partition-Space Random Walks*

Lipkovich (2002) used a Metropolis algorithm to explore a space of partitions, assuming class-specific multivariate normal or linear regression models. According to his algorithm, each Metropolis proposal is a change of class assignment for a single unit. In this section we will describe the algorithm in greater detail. First, since we will explore some variations on the idea, the following definition seems useful.

**Definition** (Partition Space Random Walk). We will define a partition space random walk to be an MCMC method generating a sample of partitions, where each is generated by a random modification of the preceding.

This suggested term emphasizes a relationship to random walk Metropolis as used for exploring a parameter space (Robert and Casella, 2004).

According to the algorithm suggested by Lipkovich, the Metropolis proposal is a change in class assignment for a single unit, selected at random. For a model with more than two classes, the new class is selected at random as well. To implement a minimum cluster size, the proposal may be repeated until a unit is selected from a cluster with size larger than the minimum.

The likelihood associated with a classification is of the classification form (1.1), optimized over nuisance parameters. A penalty is added to the log-likelihood, similar in form to the Schwarz approximation, but computed separately for each cluster and summed over clusters. For the Metropolis acceptance probability a ratio of penalized, maximized likelihoods is computed and used in computing a Bayes factor.

The approach can be rationalized from a non-Bayesian viewpoint. Considering how the BIC tends to be used in practice -- for model selection -- the approach may be viewed as a search for a model with a good balance between fit and parsimony. The method can be viewed in Bayesian terms as implementing a model where each classification has equal probability *a priori*. (Then detailed balance equations (1.3) are satisfied for each pair of classifications.)

Lipkovich reports improved performance when stations that belong to the same river basin are required to cluster together. With the suggested modification, the proposal step involves random selection of a basin, which may be re-assigned depending on the outcome of the acceptance step.

#### *1.2.4 Gibbs and other Block-at-a-Time Strategies with Latent Information*

Block-at-a-time strategies, such as the Gibbs sampler, are important tools of posterior sampling, and are particularly relevant for modeling with unknown classifications. The idea is that the unknowns for a model, or blocks of unknowns, are updated in turn. The unknowns may be updated in fixed or random order. When updating a given unknown, the values are held fixed for other unknowns.

The approach can be valuable particularly for implementing models that can be formulated in terms of “latent” (unobserved) random variables. Examples of such models include random effects models, models for censored or truncated observations, and factor analysis models, as well as mixture and classification models. Simplifications can be exploited, that result from fixing values of some random unknowns. For our models in

particular, fixing the classification can result in a standard Bayes linear model with a categorical variable identifying the class, as in an ANCOVA. Posterior sampling for such a model may be straightforward.

The block-at-a-time approach involves the concept of a full conditional distribution. Suppose we enumerate  $q$  unknowns with values  $U'_1, \dots, U'_q$  at the current step, and that a proposal is  $U'_1, \dots, U'_{q-1}, U''_q$  (the proposal changes only one of the unknowns). Then the relative odds favoring the proposed model relative to the current model is

$$\text{RO} = \frac{\pi(U'_1, \dots, U'_{q-1}, U''_q \mid \mathbf{D})}{\pi(U'_1, \dots, U'_{q-1}, U'_q \mid \mathbf{D})} = \frac{\pi(U''_q \mid U'_1, \dots, U'_{q-1}, \mathbf{D})}{\pi(U'_q \mid U'_1, \dots, U'_{q-1}, \mathbf{D})}.$$

where  $\pi(* \mid *, \mathbf{D})$  denotes a conditional posterior density and  $\mathbf{D}$  represents the data. On the right hand, the RO is computed from a ratio of full conditional densities, which are densities associated with single unknowns, fixing values for other unknowns. (Regarding our notation, we may add primes to symbols to indicate current values in an iterative scheme. Multiple primes are added to a symbol wherever we need to distinguish successive updates.)

An important case is the Gibbs sampler. When the proposal for a given unknown is a draw from the corresponding full conditional, it happens that the acceptance probability computed using the general expression equals 1, so the proposal is accepted. Thus proposal and acceptance steps are not explicit. The approach is simply to cycle through unknowns drawing a value of each from its full conditional, with values for other unknowns fixed.

In many latent-data situations the Gibbs sampler competes with the expectation-maximization (EM) algorithm. Some practitioners combine the EM with the Gibbs sampler, by starting chains at optima found by an EM algorithm, with random initializations (Gelman *et al.*, 2003; Viele and Tong, 2002).

### 1.2.5 *Non-Bayesian Simulation in Cluster Analysis*

Our study emphasizes procedures that generate random samples of classifications, from a Bayesian posterior. Many non-Bayesian simulation procedures are available, which are likely to be useful the context of modeling with classifications (Dudoit and Fridlyand, 2002; MacLachlan and Peel, 2000; Monti *et al.*, 2003; Qin and Self, 2006; Rocke and Dai, 2003). While these methods are based on various conceptual frameworks, some computational problems in common relate to representing and manipulating samples of classifications (see Chapter 2).

## 1.3 CLASS-SPECIFIC MODELS

### 1.3.1 *Multivariate Normal and $t$ Distributions*

An important form of model-based clustering involves extension of multivariate procedures, often based on multivariate normality. “Mixtures” of normal and  $t$  distributions have been studied by numerous authors. An importance difference between these models and the class regressions that we use is a requirement for distributional assumptions for all variables. With class regression models we treat some variables – our regressors – as fixed.

The package `mclust` (Fraley and Raftery, 1998, 2002, 2005) implements clustering based on mixtures of multivariate normal distributions. A useful family of models is obtained by defining parameters representing cluster “volume,” “shape,” and “orientation,” based on spectral representation of the covariance matrix (Banfield and Raftery, 1993; Celeux and Govaert, 1995). The BIC is used to choose a model, defined in terms of covariance structure and number of classes.

Computational procedures that have been used for these models include EM for mixture likelihoods (MacLachlan *et al.*, 2000) and Gibbs sampling for Bayesian analysis (Bensmail *et al.* 1997; Ter Braak *et al.*, 2002). The package `EMMIX` (MacLachlan *et al.*,

2000) provides clustering based on class multivariate  $t$  distributions, with a CEM implementation.

Ter Braak *et al.* (2002) provide an ecological application with class multivariate normal distributions. They specifically argue for multivariate distributions, with distributional assumptions for predictor as well as response variables, but mention that class-specific regressions may be convenient for modeling nonlinear relationships.

### *1.3.2 Linear and Nonlinear Regression Models*

Many authors have studied mixture models with class-specific linear or generalized linear models. Implementation may be based on maximum likelihood by the EM algorithm (DeSarbo and Cron, 1988; McLachlan and Peel, 2000; Wedel and Kamakura, 2000) or on a Bayesian model implemented using a Gibbs sampler (*e.g.*, Viele and Tong, 2002).

Our applied interest is in detecting relationships between ecological quality and ecological stressors, relationships that may be specific to a geographic region. Therefore we study models that incorporate class-specific regressions. Also, the relationships of interest to us are likely to be nonlinear. By using class-specific semiparametric methods we allow for relatively general, possibly nonlinear relationships.

### *1.3.3 Semiparametric Regression Models with a Single Predictor*

Relevant information for choice of a class-specific family of regression models may include the possibility of thresholds or non-monotone relationships, and so on. However, in view of limitations of the available information, we may be interested in relatively general families of functions. We draw on methods of semiparametric regression. We refer frequently to the treatment given by Ruppert, Wand, and Carroll (2003). Other useful, general treatments include Eubank (1988), Fan and Gijbels (1996), and Hastie and Tibshirani (1990).

It will be feasible to discuss only a few semiparametric methods. We focus particularly on two approaches that seem particularly amenable to incorporation in our model-based

clustering approach. A basis function approach allows straightforward extension of methods based on general linear models (Denison *et al.*, 2002; Fahrmeir and Tutz, 2001). An approach based on random walk priors has a close relationship to penalized least-squares and is amenable to Gibbs sampling.

For the basis function approach, the fitted curve  $\hat{f}$  is the projection of the response vector in the span of a set of basis vectors that -- as when fitting a polynomial -- may include multiple functions of any predictor. Various useful basis systems are available. An appropriate system can be used to fit a function that is continuous and piecewise linear, or piecewise polynomial, and so on. A continuous, piecewise linear function can be defined by a set of knots (breakpoints), say  $\kappa_1, \dots, \kappa_m$ , such that the function is linear below  $\kappa_1$ , between  $\kappa_1$  and  $\kappa_2$ , so on. We may specify a function that is piecewise polynomial, say of degree  $p$ , on intervals demarcated by a finite set of knots. For a straightforward implementation, we may simply regress the response variable on a set of truncated power functions. The design matrix has  $i$ th row given by

$$\left(1, x_i, \dots, x_i^p, (x_i - \kappa_1)_+^p, \dots, (x_i - \kappa_m)_+^p\right) \quad (1.8)$$

where  $x_i$  denotes the value of the predictor for the  $i$ th unit. Here the notation  $f_+$  denotes a function that is equivalent to function  $f$  except that  $f_+(x) = 0$  for  $x$  such that  $f_+(x) < 0$ . The fitted curve obtained by this approach has derivatives up to order  $p - 1$ .

It is clear that that the basis functions are correlated, so that multicollinearity should concern us. Orthonormal basis functions may be preferred, a common choice being  $B$ -splines (Eilers and Marx, 1996; Henderson, 2006). (However, regression on truncated power functions is developed extensively by Ruppert *et al.*)

The approach based on splines may require care in selection of knots. Automated schemes for knot selection are reviewed in Ruppert *et al.* Alternatively, a common practice is to use a large number of knots and avoid over-fitting by incorporating a

roughness penalty. The function optimized has a penalty term added to a sum of squared residuals. An example from Fahrmeir and Tutz (2001, Expressions 5.4-5.6) is

$$\hat{f} = \arg \min_f \left\{ \sum_{i=1}^n (y_i - f_i)^2 + \lambda \sum_{i=3}^n (f_i - 2f_{i-1} + f_{i-2})^2 \right\} \quad (1.9)$$

For this particular penalty, observe that  $(f_i - 2f_{i-1} + f_{i-2})^2$  is proportional to the squared difference between a function value and the average of two flanking function values, when the function is evaluated for equally spaced points in the domain.

Alternative penalties are discussed in Ruppert *et al.*, who particularly emphasize penalties related to ridge regression.

A general feature of the penalization approach is introduction of a roughness parameter  $\lambda \geq 0$ . Setting  $\lambda$  to zero eliminates the penalty and results in interpolation, while setting  $\lambda$  to a large value results in a global least-squares polynomial fit. Choosing an appropriate degree of smoothness involves a tradeoff between bias and variance: Bias may be reduced by using less smooth curves, while more smooth curves may provide lower variance. The problem of selecting knots, in the basis function approach, is now replaced by a problem of specifying a degree of smoothness.

Use of (1.9) corresponds to computing posterior means under a particular Bayesian model. Suppose our prior for the fitted curve derives from the second-order Gaussian random walk

$$\begin{aligned} f_i \mid f_{i-1}, f_{i-2} &\sim N(2f_{i-1} - f_{i-2}, \tau^2), \quad i > 2, \\ f_{i-1}, f_{i-2} &\sim N(0, \sigma_0^2), \end{aligned}$$

and let the  $i$ th observation be distributed  $N(f_i, \sigma^2)$ , so that our prior has parameters  $\sigma_0^2, \sigma^2, \tau^2$  (Fahrmeir and Tutz, 2001). With equally spaced predictor values and one observation for each, (1.9) gives the posterior means, with  $\lambda = \sigma^2 / \tau^2$ . However we may incorporate uncertainty in the appropriate degree of smoothing. It is suggested that  $\sigma^2$  and  $\tau^2$  be assigned (hyper)priors, say inverse gamma distributions, resulting in a



Bayes hierarchical model.  $\sigma_0^2$  may be set to some large value. A Gibbs sampler with analytic full conditionals is available.

#### 1.3.4 *Semiparametric Regression with Multiple Predictors*

For the ecological contexts of interest to us, there will be multiple predictors, more often than not. Accordingly, suppose we have  $p > 1$  predictors. Several semiparametric approaches are available. Again, Ruppert *et al.* (2003) provide a useful overview.

- 1) Sometimes an additive model (Hastie and Tibshirani, 1990; Venables and Ripley 1998) may be useful. The fitted function is simply a sum of smooth, predictor-specific functions, so that any interactions are neglected. Yuan and Norton (2003) provide an important application to ecological risk assessment.
- 2) In a basis function approach we may include additional basis vectors to represent interactions. The usual approach for incorporating interactions in a linear model – involving element-by-element products of design matrix columns -- is termed the *tensor product* approach. Lamon and Clyde (2000) provide an interesting ecological application.
- 3) Also within a basis function approach, we may rely on some special system of  $p$ -dimensional basis functions such as the radial basis functions (Ruppert *et al.*).
- 4) We may model interactions based on a multidimensional stochastic process, an approach conventional in geostatistics.

Lamon and Clyde (2000) combine a tensor product approach with Bayesian model averaging, to relate Chlorophyll *a* concentration to nutrient concentrations and other predictors, for Florida's Lake Okeechobee. (Chlorophyll *a* concentration is a correlate of biomass of algae that may pose a threat to water quality.) The semiparametric approach used by Lamon and Clyde allows for nonlinearities and interactions. Basis vectors for the approach are truncated power functions and tensor products. A Bayesian model averaging approach (Smith and Kohn, 1996) is used to avoid over-fitting, at the same time allowing for model uncertainty.

We suggest that a possible shortcoming of a tensor product approach relates to economical representation of thresholds, in case of multiple stressors. Consider the case of exposure to multiple toxins, each with a special toxic threshold concentration. For a single toxin, a sub-threshold exposure may not result in any observable effect. Such a relationship is easily represented by incorporating a break-point in the dose-response function. However, economical representation of a multidimensional threshold, in case of simultaneous exposure to multiple toxins, may be difficult.

Also, the tensor product approach introduces a dependence on the coordinate system in which the predictors are reported. For rotational invariance, Ruppert *et al.* prefer an approach based on radial basis functions.

#### 1.4 THE LABEL-SWITCHING PROBLEM AND TWO TYPES OF SOLUTIONS

We use simulation algorithms to sample a posterior. The posterior sample must be summarized, using summary statistics or graphs. The task of useful summarization is complicated by the label-switching problem that we introduced in Chapter 1.

In some applications, the problem is avoided because of natural constraints on model unknowns, which result in an identified model. For example, when searching for a small cluster in a background of noise, we can define  $C_1$  as the small cluster and  $C_2$  as a background cluster. Also, in a supervised classification context, each class is identified with one or more units in a “training” set.

A conventional approach has been to impose prior order constraints on parameters. Class means may be required to satisfy  $\mu_1 < \dots < \mu_k$ , or class probabilities to satisfy  $\tau_1 < \dots < \tau_k$ . However, it is easy to imagine situations where any simple constraint will fail to adequately describe the likelihood surface (Celeux *et al.*, 2000; Stephens, 2000; Frühwirth-Schnatter, 2001). Even for a simple location mixture, one can imagine a 3-class situation with parameter values equal in each class, except for a distinct mean for one class, and a distinct class probability for another. If we rely on parameter order constraints, the results could differ, depending on whether the constraint is on class

means or class probabilities. Selection of useful parameter constraints is inevitably more difficult if the class-specific models are multivariate or semi-parametric.

In Chapter 2 we investigate two types of approach, both with some background in cluster analysis literature. Various authors post-process the sample, adjusting class indices to optimize some measure of agreement among classifications. The term *relabeling* is used in the literature of “finite mixture” models (Celeux *et al.*, 2000; Qin and Self, 2006; Stephens, 2000). A term suggested by Lipkovich (2002) is *alignment*, invoking an analogous problem encountered in factor analysis (Clarkson, 1979; Ichikawa and Konishi, 1995). Relabeling -- or alignment -- may be necessary for computation of class-specific summaries or estimates.

Various relabeling procedures may be plausible in a given situation, associated with different model unknowns. Much of model-based clustering is based on “finite mixture” models, for which one standard computational approach is a Gibbs sampler (*e.g.*, Bensmail *et al.*, 1997; Diebolt and Robert, 1994; Viele and Tong, 2002; Wasserman, 2002). In this context, Stephens (2000) suggests relabeling based on comparison of membership probabilities, as computed routinely for the Gibbs sampling approach. Alternatively, relabeling may be based on class-specific parameter estimates (Celeux *et al.*, 2000; Qin and Self, 2006). Lipkovich (2002) suggests adjusting class labels to maximize some agreement among results from different MCMC steps. We present in Chapter 2 a relabeling algorithm that maximizes a particularly simple measure of agreement. We follow Lipkovich in emphasizing manipulations of classifications. Such an approach may be used with diverse Monte Carlo procedures.

A second approach avoids relabeling by reliance on a particular label-invariant statistic. Monti *et al.* (2003) used the term *consensus matrix* to describe an  $n \times n$  matrix that gives, for each pair of units, the sample fraction with the two units assigned to the same cluster. The primary interest of Monti *et al.* related to the use of subsampling to evaluate stability of classifications. They also discuss applications for evaluating the degree of support for clusters, for ranking units according to their value for representing particular

clusters, and for estimating the number of clusters. Monti *et al.* did not relate their approach to the label-switching problem, and did not actually mention the important property of label-invariance. Tibshirani *et al.* (2001) develop a cross-validation approach based on similar ideas.

We enlarge on the applications of the consensus matrix, particularly as a basis for visualization techniques that do not require relabeling. Reflecting our use in the posterior sampling context we use the term *estimated co-clustering probabilities* (ECCP). In particular, we use theory from multidimensional scaling to develop visualization procedures.

## Chapter 2

### Solutions to the Label-Switching Problem using Relabeling and Label-Invariant Summarization

Our procedures assume an unknown classification of  $n$  units into  $k$  disjoint subsets. Our units are monitoring stations or groups of nearby monitoring stations. Bayesian procedures are used to evaluate uncertainty in the classification. We implement the approach by sampling the posterior using Markov chain Monte Carlo (MCMC).

As discussed in Chapter 1, the procedure is subject to the label-switching problem. In our situation the problem is a form of model non-identification, due to a likelihood that is invariant under alternative labeling of classes. The problem will be particularly important when, as in our applications, the posterior sample is obtained by pooling multiple chains generated using MCMC. In Chapter 1 we introduced two general types of solution, one based on relabeling and a second emphasizing reliance on label-invariant summaries. We now develop in detail procedures of each type.

For our label-invariant approach, we rely on a matrix of ECCP described in the previous chapter (also see Monti *et al.*, 2003). The matrix is the basis for useful visualization procedures that reveal the pattern of clustering and the confidence in assigning units to clusters. We provide a justification for evaluating the matrix using the method of principal coordinates, a particularly straightforward approach to the problem of multidimensional scaling. Indeed, we observe that when the ECCP matrix is transformed as customary for principal coordinates analysis (Gower, 1966), the result is a non-negative definite matrix, as assumed by that procedure. Euclidean distances among units in this representation have a simple relationship to the ECCP.

Several relabeling procedures have been proposed that might be adapted to our applications (Celeux, Hurn, and Robert, 2000; Qin and Self, 2006; Stephens 2000). We

suggest that an algorithm that operates on the sample of classifications, as sketched by Lipkovich (2002) is likely to have some advantages. In particular, storage of quantities other than the classification requires much more storage and may not always be necessary, as in an example in the following chapter, with semiparametric class regressions. Our approach is relatively general in view of the possibility of application with the various Bayesian and non-Bayesian simulation procedures that generate random samples of classifications. Some information is given on computing time for our algorithm.

Our term *estimated co-clustering probability* (ECCP) reflects an emphasis on Bayesian posterior sampling. For reasons that will become evident, a term that may be appropriate for more general applications is *sample concurrence fractions*.

The study is organized as follows. In Section 2.1 we present some terminology and matrix theory. Label-invariant summarization based on concurrence fractions is the topic of Section 2.2, while Section 2.3 presents our approach to relabeling. In Section 2.4 we attempt an integration of the procedures with other post-processing tasks, in the context of MCMC with multiple chains. Some technical results are presented in chapter appendices.

## 2.1 PARTITIONS AND CLASSIFICATIONS, WITH MATRIX REPRESENTATIONS

A distinction from Chapter 1 is between a partition and a classification. A partition can be specified by stating, for each pair of units, whether or not both fall in the same cluster. A classification is obtained from a partition by assigning labels to the clusters. We point out matrix representations of these objects.

Borrowing terminology from experimental design, a classification may be represented by an  $n \times k$  *incidence* matrix  $\mathbf{Z}$ . The value in the  $i$ th row and  $j$ th column equals 1 if the  $i$ th unit belongs to the  $k$ th cluster, according to a specific choice of class labels, and otherwise equals zero. A partition may be represented by a *concurrence* matrix  $\mathbf{M}$ , a

symmetric  $n \times n$  matrix where the value in the  $i$ th row and  $j$ th column is 1 if units  $i$  and  $j$  belong to the same cluster, and otherwise zero. ( $\mathbf{M}$  is termed a *connectivity matrix* by Monti *et al.*, 2003). By the definition of a partition,  $\mathbf{M}$  can be arranged into block-diagonal form by permuting the order of units in some way.

Associated with a concurrence matrix  $\mathbf{M}$  there are  $k!$  incidence matrices associated with possible classifications, identical up to a permutation of the order of columns. Indeed, for a concurrence matrix  $\mathbf{M}$ , suppose  $\mathbf{Z}^*$  is an incidence matrix derived by permuting columns of another incidence matrix  $\mathbf{Z}$ , so that both represent the same partition. Then  $\mathbf{Z}^* = \mathbf{Z}\mathbf{B}$  where  $\mathbf{B}$  is a permutation matrix. Using the orthonormal property of  $\mathbf{B}$ , we have

$$\mathbf{Z}^*\mathbf{Z}^{*\text{T}} = \mathbf{Z}\mathbf{B}(\mathbf{Z}\mathbf{B})^{\text{T}} = \mathbf{Z}\mathbf{B}\mathbf{B}^{\text{T}}\mathbf{Z}^{\text{T}} = \mathbf{Z}\mathbf{Z}^{\text{T}} = \mathbf{M}.$$

Properties of permutation matrices are discussed in sources such as Harville (1997).

## 2.2 A LABEL-INVARIANT APPROACH TO POST-PROCESSING

### 2.2.1 The ECCP matrix

It is convenient to arrange the ECCP in the form of a symmetric matrix, **ECCP** say.  $\text{ECCP}_{ij}$  will denote the value in the  $i$ th row and  $j$ th column, equal to the sample fraction with the  $i$ th and  $j$ th units falling in the same cluster. The matrix is symmetric with diagonal values equal to unity.

Suppose that some Monte Carlo scheme such as resampling or Bayes posterior sampling generates a random sample of classifications. Let  $\mathbf{M}_1^*, \dots, \mathbf{M}_{n_{\text{MC}}}^*$  denote concurrence matrices for a sample of classifications, of size  $n_{\text{MC}}$ . Then

$$\mathbf{ECCP} = \frac{1}{n_{\text{MC}}} \sum_{s=1}^{n_{\text{MC}}} \mathbf{M}_s^*.$$

Like the concurrence matrices from which it is computed, **ECCP** does not depend on the labeling of particular classifications in the sample. In the remainder of this section we outline several applications.

### 2.2.2 *Evaluating the ECCP Using Clustering and Ordination*

The ECCP are affected by how classifications in our sample are labeled, and provide a basis for label-invariant visualization techniques. Useful graphs can be based on general techniques for display of similarity and dissimilarity information, namely cluster analysis and multidimensional scaling (Mardia *et al.*, 1977; Seber *et al.*, 1984). In this context the ECCP are treated as a kind of similarity. Where a procedure requires a measure of dissimilarity we use

$$d_{ij} = 1 - \text{ECCP}_{ij}, \quad (2.1)$$

interpreted as the sample fraction with the *i*th and *j*th units assigned to different classes. The quantities (2.1) play an important role in two graphs that we use, an average-linkage dendrogram, and a plot of principal coordinates. Use of a dendrogram is mentioned by Monti *et al.* (2003). (However, those authors emphasize direct display of the matrix with different colors or shading used to convey relative magnitudes.)

The method of principal coordinates (Gower, 1966) is a particularly straightforward solution of the problem of ordination or multidimensional scaling (MDS), encapsulated essentially in Theorem 1 in the chapter appendix. The general MDS problem is to evaluate a matrix of similarities by finding a correspondence between the *n* units and coordinates in an *n*-dimensional Euclidean space, such that distances have a definite, ideally simple relationship to the similarities. Specific problems include identifying the properties that a measure of similarity should satisfy, and computing coordinates for plotting the units.

Principal coordinates is an MDS method applicable when a non-negative definite matrix results from a particular transformation to the similarity matrix. Then the plotting coordinates are based on eigenvectors of the transformed matrix. We observe (Corollary 1) that we do obtain a non-negative definite matrix using the appropriate transformation



of our matrix **ECCP**. Squared Euclidean distances generated by the method are proportional to dissimilarities of the form (2.1).

### *2.2.3 Label-Invariant Identification of Clusters from Simulation Output, with Evaluation of Classification Uncertainty*

Our results will include a classification of the units. Also, we may provide some evaluation of the confidence in assigning particular units. The most obvious approach requires that we relabel the sample: We may compute the sample frequency with a given unit belonging to each class, and assign the unit to the class that includes it with highest frequency. A measure of uncertainty is obtained by subtracting the maximum membership probability for a unit from one (Fraley and Raftery, 2005; Lipkovich, 2002).

We emphasize label-invariant graphical techniques, described in the previous section, that do provide some sense of uncertainty in classification. For a label-invariant computation of the classification, we generate a dendrogram from the ECCP using the average-linkage approach, and extract clusters by “cutting” the dendrogram at an appropriate height. The procedures are automated using R library functions: We use the function `hclust` to generate the dendrogram `cutree` to extract clusters.

In Chapter 3, we will give a plausible index of uncertainty for co-clustering of units  $i$  and  $j$ , which is found to show some promise in the context of selecting the most appropriate number of classes.

### *2.2.4 Conditional Estimation of Class-Specific Parameters*

In addition to inference of a classification, we may generate inferences related to class-specific parameters, in our applications parameters of class regressions. A naive approach is to estimate class-specific parameters with the classification fixed, treating an estimated classification as if known to be the true classification. Based on cluster analysis literature, it seems that such a conditional estimate may be biased, tending to overestimate differences among groups (Gordon, 1966). A more sophisticated approach

may involve an evaluation of joint uncertainty for the classification and other model unknowns. However, the conditional approach may be practical in some situations.

### 2.3 RELABELING AND RELATED COMPUTATIONS

Relabeling may be desirable for computation of class-specific summaries from simulation output, for example if we compute the sample fraction with a particular unit assigned to a particular class. In this section we first develop an “alignment criterion,” to be maximized by adjustment of class labels, then present a relabeling algorithm.

#### 2.3.1 The Alignment Matrix

In developing the alignment criterion, we first consider comparison of two classifications. It is helpful to consider a particular tabular comparison of two classifications, which we illustrate by comparing two classifications of the same 5 units into 2 groups. Let two classifications be  $P = (1, 1, 1, 2, 2)$  and  $Q = (2, 2, 2, 1, 2)$ , where classification is represented by a vector with the  $i$ th coordinate giving the class index for the  $i$ th unit. It is useful to cross-classify the units according to class labels under  $P$  and  $Q$ :

**Table 2-1** Alignment matrix for hypothetical example.

Cluster Index in $P$	Cluster Index in $Q$	
	1	2
1	0	3
2	1	1

Here the cell count in the  $i$ th row and  $j$ th column,  $n_{ij}$  say, is the number of units assigned to the  $i$ th class under  $P$  and to the  $j$ th class under  $Q$ . We suggest calling such a table an *alignment matrix*.

The alignment matrix is apparently useful for multiple computations. Indices that express the agreement between partitions – not necessarily labeled -- are computed from the matrix (Rand, 1971; Hubert and Arabie, 1985). We suggest that the matrix trace, which gives the count of units with the same label under each classification, can be taken as a quantity to maximize in a relabeling algorithm.

### 2.3.2 Matrix Representation of a Criterion for Aligning Two Classifications

For mutual alignment of two classifications, our suggested approach is to maximize the trace of the alignment matrix. The suggested criterion is related to the index of Cohen (1960), which quantifies inter-observer reliability in recording a categorical variable.

Let two classifications of the same units be represented by incidence matrices  $\mathbf{Z}_1$  and  $\mathbf{Z}_2$ . We first observe that the alignment matrix can be expressed as  $\mathbf{Z}_1^T \mathbf{Z}_2$ . We suggest maximizing the trace

$$A = \text{tr}(\mathbf{Z}_1^T \mathbf{Z}_2) \quad (2.1)$$

In maximizing this quantity, we will adopt a convention of holding the labels fixed for the first classification and varying the labels for the second. For a matrix representation, we may let  $\mathbf{B}$  denote a permutation matrix, and maximize

$$A(\mathbf{B}) = \text{tr}(\mathbf{Z}_1^T \mathbf{Z}_2 \mathbf{B}).$$

(While we use permutation matrices to complete our matrix representation, use of such matrices in computer code would be inefficient.)

For our example, if we fix the labels for  $P$  and swap the class labels for  $Q$ , the recomputed alignment matrix is  $\begin{pmatrix} 3 & 0 \\ 1 & 1 \end{pmatrix}$ . We see that the matrix trace has now increased in value from 1 to 4, which may be taken to support swapping the class indices in one classification.

### 2.3.3 A Criterion for Alignment of a Sample of Classifications

The approach for comparing two classifications can be generalized to provide a criterion to maximize, in relabeling a sample with two or more classifications. In our approach, each classification is relabeled based on comparison to a summary of other classifications, where the latter are treated as mutually aligned. The approach may be iterated.

Consider alignment of a sample of a classifications with incidence matrices  $\mathbf{Z}_1, \dots, \mathbf{Z}_{n_{MC}}$  according to an initial labeling. Without loss of generality, consider alignment of the first classification with incidence matrix  $\mathbf{Z}_1$ . A useful generalization of the alignment

matrix is  $\left( \sum_{j=2}^{n_{MC}} \mathbf{Z}_j \right)^T \mathbf{Z}_1$ . We again maximize the trace over column permutations, *i.e.*, choose a permutation matrix  $\mathbf{B}$  maximizing

$$\text{tr} \left( \left( \sum_{j=2}^{n_{MC}} \mathbf{Z}_j \right)^T \mathbf{Z}_1 \mathbf{B} \right). \quad (2.2)$$

For an iterative approach, (2.2) is maximized on a given iteration, using incidence matrices carried forward from the previous iteration.

An objective function that may be optimized by such a procedure, apart from local optima, is

$$\sum_{i \neq j} \text{tr} (\mathbf{Z}_i^T \mathbf{Z}_j),$$

the sum of values given by Expression (2.1) over pairs of classifications. An iteration of our approach will not decrease the value of this function. Indeed, a single iteration will increase the objective function by a value equal to the increase of (2.2).

### 2.3.4 A Relabeling Algorithm

Our suggested algorithm maximizes (2.2), with initial labeling based on the ECCP dendrogram in Section 2. The steps of the suggested algorithm are:

- (1) Use a label-invariant procedure to select  $k$  cluster representative units (CRU), one for each class,  $\text{CRU}_1, \dots, \text{CRU}_k$ .
- (2) Use the CRU from (1) in a preliminary relabeling of each classification, where possible assigning index  $j$  uniquely to the class that includes  $\text{CRU}_j$ .
- (3) Adjust the labels for each classification so as to maximize (2.2).

Step (3) could be iterated. However, in our experience the initial labeling is very effective. Therefore we execute Step (3) only once.

Some additional details are as follows. For selecting CRU in Step (1), a general principle is to choose  $k$  units that with high probability represent all  $k$  classes. We choose CRU minimizing the sum over cells in the corresponding  $k \times k$  submatrix of **EC**CP. That sum is evaluated for each possible choice of  $k$  units. In case of a tie we use the first selection in our enumeration of selections that achieves the minimum.

In Step (2) a cluster may contain more than one of the CRU, so that other clusters contain none. In that case the labeling of the classification is not modified. In Step (3) we rely on exhaustive evaluation of the  $k!$  label permutations. This computation is reasonably efficient in our applications, which use 2-4 classes, particularly given that the alignment table does not need to be recomputed for each label permutation. For the particular case of 2 classes, the optimal label permutation is found simply by noting whether or not the trace of the alignment matrix is larger than the sum of values for non-diagonal cells.

Our relabeling algorithm follows Stephens (2000) in that relabeling is based on a loop through the sample, with each item in the sample compared to a summary of the rest of the sample. However, whereas in Stephens' algorithm comparisons are based on membership probabilities, routinely generated using a standard Gibbs sampler for "finite mixture" modeling, our approach relies on comparisons among classifications.

A preliminary evaluation of our relabeling algorithm was performed for 2-4 class models, for the analysis of Maryland dataset, as described in the next chapter. Our relabeling algorithm was timed in Windows XP on a 1.6 Gigahertz Pentium processor. For 2-4 classes, the initial labeling required 6-10 seconds per 10,000 random classifications, while the refinement in the final step required 10-20 seconds per 10,000. Generally the computational expense is somewhat larger with increasing  $k$ . Such computing times are negligible relative to the hours of computing required for posterior sampling, but can still

represent a nuisance in some situations. Relabeling may have been relatively rapid in our experiment due to a relatively small number of units and classes. Also, we store relabeling results without rearranging objects -- matrices etc. -- that represent the unlabeled sample. The latter approach would increase execution time.

We find that the result of our initial labeling is often close to our final result. The initial labeling may be more effective with a smaller number of classes. The fraction of classifications revised in our Step (3) was 0 for 2 classes, fewer than 1 per thousand for 3 classes, and 1% for 4 classes.

## 2.5 DISCUSSION

We currently use two graphs for label-invariant evaluation of simulation output, a dendrogram (also see Monti et al., 2003) and an ordination plot that represents a novel application of multidimensional scaling. Each plot may have particular advantages. Dendrograms can be associated with certain distortions of information on similarities (Everitt, 1993; Seber, 1984) but are helpful for displaying the most likely classification, along with some indication of uncertainty. However, for units with relatively uncertain class assignment, our ordination plot seems helpful in displaying information on the proximity of a unit to each cluster (not only to its assigned cluster). As in other cluster analysis situations it may be desirable to examine results from more than one procedure.

Proposed relabeling procedures can be said to operate in three “spaces.” The procedure that we suggest is based on comparison of classifications and can be said to operate in “classification space” (see also Lipkovich, 2002). Alternatively, each item in the sample includes values of class distribution parameters, and these can also serve as a basis for relabeling (Celeux et al., 2000; Qin and Self, 2006). Finally, Stephens (2005) suggests relabeling based on comparison of class membership probabilities as computed for a “finite mixture” model. Some simplification of computations is obtained by choosing one item in the sample as a reference, perhaps associated with the maximum likelihood value, and aligning other items based on comparisons to the reference (Lipkovich, 2002;

Qin and Self, 2006). It seems that the multiplicity of reasonable relabeling approaches contributes to the appeal of label-invariant procedures.

It is desirable to conduct quantitative comparisons of alternative relabeling procedures. Procedures may be compared with respect to computational efficiency or effects on inference. We do not know of evaluations of sampling properties of cluster analysis procedures that incorporate relabeling.

Absent quantitative comparisons, we observe that large amounts of computer memory may be required for storage of parameter values or class membership probabilities, required for the alternative relabeling procedures. For applications with semiparametric class regression models we do not store the class regression parameters for all simulations, which would require approximately 20 megabytes of storage for one of the simulations we report. Storage of class membership probabilities would be even more expensive. Our computations do not require explicit class probabilities, as used for a pure Gibbs sampler. Considering the range of Bayesian and frequentist procedures that generate random samples of classifications, a classification-space approach may be extended more easily than alternative approaches.

## 2A APPENDIX: TECHNICAL RESULTS RELATED TO THE ORDINATION APPROACH

The following statement of the method of principal coordinates follows Seber (1984).

**Theorem 1. Principal Coordinates Analysis of a Matrix of Similarities.** Let  $\mathbf{C}$  denote a matrix of similarities among  $n$  units, with typical element  $c_{ij} \in [0, 1]$ , subject to

$c_{ii} = 1$  ( $i = 1, \dots, n$ ). Define  $\mathbf{F} = (\mathbf{I} - \bar{\mathbf{J}}) \mathbf{C} (\mathbf{I} - \bar{\mathbf{J}})$  where  $\mathbf{I}$  is an identity matrix and  $\bar{\mathbf{J}}$  is a square matrix with each value equal to  $1/n$ , and define

$\mathbf{G} = (\gamma_1^{1/2} \mathbf{v}_1 \vdots \dots \vdots \gamma_p^{1/2} \mathbf{v}_p)$  where  $\gamma_1, \dots, \gamma_p$  are positive eigenvalues and  $\mathbf{v}_1, \dots, \mathbf{v}_p$

corresponding eigenvectors in the spectral representation of  $\mathbf{F}$ . If  $\mathbf{F}$  is non-negative definite, the  $i$ th row of  $\mathbf{G}$  gives coordinates for the  $i$ th unit ( $i = 1, \dots, n$ ) in  $n$  dimensions,

with the squared Euclidean distance between coordinates for the  $i$ th and  $j$ th units equal to  $2(1 - c_{ij})$ .

**Corollary 1. Principal Coordinates Analysis of ECCP.** When **ECCP** is transformed for principal coordinates analysis as described in Theorem 1, the result is a non-negative definite matrix. Rows of the matrix give coordinates of corresponding units, separated by squared Euclidean distances  $2(1 - \text{ECCP}_{ij})$ .

**Proof.** ECCP is a sample average of concurrence matrices,  $\mathbf{M}_1^*, \dots, \mathbf{M}_{n_{MC}}^*$  say. We observe that any concurrence matrix  $\mathbf{M}$  is subject to a standard factorization from experimental design,  $\mathbf{M} = \mathbf{Z}\mathbf{Z}^T$  where  $\mathbf{Z}$  is an incidence matrix. Therefore

$$(\mathbf{I} - \bar{\mathbf{J}}) \mathbf{M} (\mathbf{I} - \bar{\mathbf{J}}) = (\mathbf{I} - \bar{\mathbf{J}}) \mathbf{Z} [(\mathbf{I} - \bar{\mathbf{J}}) \mathbf{Z}]^T$$

for some incidence matrix  $\mathbf{Z}$ . A matrix of this form is non-negative definite (Harville, 1997, Corollary 14.2.14). Thus  $(\mathbf{I} - \bar{\mathbf{J}}) \mathbf{ECCP} (\mathbf{I} - \bar{\mathbf{J}})$  is an average of non-negative definite matrices, and as such is non-negative definite.



## Chapter 3

### **Ecological Applications of Model-Based Clustering with Class-Specific Linear and Semi-Parametric Regressions**

We apply model-based clustering methods that infer an unknown classification of monitoring stations, with class linear regressions. Such an approach can be used to detect regional variation in the relationship of an ecological response to stressors or other predictors. Observation of regional relationships may stimulate and focus research by suggesting scientific hypotheses, for example relating to regional climate, soil, or land use. If regional variation is not taken into account, a stressor may be viewed as unimportant, when actually it has effects that are region specific. Conversely, a pattern that is apparent based on an analysis ignoring regional variation may actually be due to poorly understood regional effects.

The unknowns in our model include a classification and parameters of class regressions. We implement Bayesian inference, for convenience relying on flat priors for class-specific parameters. Alternative classifications are treated as equally probable *a priori*. We follow Lipkovich (2002) in using a Metropolis algorithm to search a space of classifications. However, whereas Lipkovich eliminated class distribution parameters by optimization, with incorporation of likelihood penalties, we implement inference jointly for such parameters, along with the classification. To sample the posterior distribution, we use a Metropolis algorithm, obtained by modification of a conventional latent data Gibbs sampler.

We also follow Lipkovich in using clustering units that are groups of nearby stations. As noted by Lipkovich, use of multiple-station units may impose useful spatial contiguity on the clustering results, and may reduce the prevalence of local optima, by reducing the size of the model space. We term these units geographic clustering units (GCU). However, while the GCU used by Lipkovich were river basins, where possible we use combinations

of river basins and ecoregions, to facilitate comparison of our clusters to standard ecoregions.

Our approach to model-based clustering is described in Section 1. In the subsequent sections we analyze two datasets, one from monitoring stations in Maryland, USA, the other representing stations in Ohio, USA. For each analysis the response variable is a multimetric index of ecological quality. Such indices have been developed by various government agencies, for evaluating the ecological quality of streams and rivers, providing an informal reduction of direct measurements on multiple variables (*e.g.*, Barbour *et al.*, 1999).

For the Maryland dataset, preliminary analyses suggest a nonlinear relationship of the response variable to the predictors. We extend a model with class linear regressions by use of *B*-splines, which we treat as additive across predictors. Although the model used for clustering does not incorporate predictor interactions, interactions can be accounted for by examining two-dimensional surfaces, fitted to data for the clusters identified.

For each analysis we compare results obtained with the number of classes fixed at different values from 2 to 4. Such a range seems adequate for detecting regional patterns on the approximate spatial scale of USEPA Level III-IV ecoregions, for datasets with the geographic extent of those considered. McMahon *et al.* (2001) suggest Level III as a suitable scale for application of statistical methods. Based on our results, we argue that the approach does probably generate useful information on regional variation, at such a scale. However, because of our small number of GCU, our analyses do not definitively address the relationship between standard ecoregions and results generated according to our approach.

All computations were programmed in R (R Development Core Team, 2006; Venables and Ripley, 1998). Important library functions include `hclust` for hierarchical clustering and `cutree` for extraction of clusters from a dendrogram object. Some contributed packages were also found useful, namely `akima` (Akima, 1978, 1996) for

multidimensional interpolation, `coda` (Plummer, 2005) for MCMC diagnostics, `gam` (Hastie, 2005) for smoothers and generalized additive models, and `mvtnorm` (Genz *et al.*, 2005) for multivariate normal random number generation.

### 3.1. BAYES MODEL AND POSTERIOR SAMPLING PROCEDURE

In this section we describe our approach to model-based clustering with class linear models. We have found it convenient to rely on a Bayesian approach, sampling the posterior using Markov chain Monte Carlo (MCMC).

#### 3.1.1 Sampling Model and Prior

Our clustering procedure assumes an unknown classification of  $n$  units into a known number  $k$  of disjoint groups  $C_1, \dots, C_k$ . Our  $n$  units, denoted  $\text{GCU}_1, \dots, \text{GCU}_n$ , are known groupings of  $m$  monitoring stations ( $m \geq n$ ).

We assume class-specific regression models. Formally,  $\mathbf{y}_i = (y_1, \dots, y_{m_i})^T$  denotes an observed response for  $m_i$  stations belonging to  $\text{GCU}_i$ . We treat  $\mathbf{y}_i$  as a realization of vector random variable  $Y_i = \mathbf{X}_i \boldsymbol{\beta}_{\gamma_i} + E_i$  where

$\mathbf{X}_i$  is a known  $m_i \times q$  matrix of predictors for stations in  $\text{GCU}_i$ ;

$\boldsymbol{\beta}_1, \dots, \boldsymbol{\beta}_k$  are class-specific coefficient vectors of length  $q$ .

$\gamma_i \in \{1, \dots, k\}$  denotes the class that includes  $\text{GCU}_i$ ;

$E_i \sim N(\mathbf{0}_{m_i}, \sigma^2 \mathbf{I}_{m_i})$  ( $i = 1, \dots, n$ ),  $E_i$  independent of  $E_j$  for  $i \neq j$ ;

$\mathbf{0}_m$  is a vector of zeros of length  $m$ ; and

$\mathbf{I}_m$  is an  $m \times m$  identity matrix.

$\mathbf{X}_i$  includes a column with all values equal to one, to account for class-specific intercepts. Predictors may be represented by single columns or, for purposes of fitting polynomial or

spline regression, etc., a single predictor variable may be represented by multiple columns, providing a basis for an appropriate linear space.

While we initially assume that the residual variance is equal for each class, a limited comparison to results allowing class-specific variances is given in the chapter discussion.

Our unknowns subject to statistical inference thus include a classification

$\gamma = (\gamma_1, \dots, \gamma_n)$  and regression parameters  $\beta_1, \dots, \beta_k, \sigma^2$ . We place constraints on the values of these unknowns, namely (1) a minimum cluster size (Diebolt and Robert, 1994; Lipkovich, 2002; Wasserman, 2000); and (2) constraints on regression coefficients, which may depend on the application. A Bayesian formulation is implemented in which each classification of GCU is treated as equally probable *a priori*, subject to a minimum number of *stations* per cluster. Our prior for  $\beta_1, \dots, \beta_k, \log(\sigma^2)$  is taken to be uniform on a subset of  $\mathbf{R}^{kq+1}$  defined by our constraints.

In our applications, the response variable is a multimetric index of ecological quality.

The values of such indices are restricted to an interval  $[0, u]$  with  $u > 0$ . We constrain expected values of the response to lie in the range of possible response values.

Specifically, for an allowed classification, and coefficient vectors  $\beta_1, \dots, \beta_k$ , we require that each element of  $\mathbf{X}_i \beta_{\gamma_i}$  ( $i = 1, \dots, n$ ) falls in  $[0, u]$ . In some cases we also impose sign constraints on the regression coefficients.

### 3.1.2 Posterior Sampling Procedure

We sample the posterior using a method of Markov chain Monte Carlo (MCMC). A straightforward approach is a Gibbs sampler that treats the classification as missing information, and alternates between updating the classification with values fixed for regression parameters, and updating regression parameters with the classification fixed. However, a variety of difficulties have been reported for such an approach, in the closely related context of “finite mixture” modeling, including instability (Viele and Tong, 2002) and inadequate exploration of the model space (Celeux *et al.*, 2000; Ter Braak *et al.*,

2003). In preliminary work, a particular Gibbs sampler worked well in some situations with classes well defined. In other cases we encountered the problem of “inefficiency in meeting constraints,” described below.

We have found it practical to adopt a missing-information Metropolis approach that alternates between updating the class assignments of single units, chosen at random, and updating of regression parameters. We implement a scheme for generating and pooling independent chains, which allows for testing convergence, and provides some confidence in exploration of the model space.

The state space for our Markov chain is combinations of  $\gamma, \beta_1, \dots, \beta_k, \sigma^2$ . Apart from the approach for updating the classification, our approach is similar to a Gibbs sampler, relying on sampling of full conditional distributions. We initialize a chain by choosing a random partition of the GCU, and setting the regression parameters equal to standard regression estimates. Then we iterate the following computations.

- (1) To update the classification  $\gamma$ , we use an explicit Metropolis proposal and acceptance, which updates the class assignment of a single unit, selected at random. The acceptance probability will depend on current values of regression parameters.
- (2) We sample  $\beta_1, \dots, \beta_k$  from a multivariate normal full conditional distribution, obtained by treating the current values for the other unknowns ( $\gamma, \sigma^2$ ) as the true values.
- (3) We sample  $\sigma^2$  from the inverse gamma full conditional distribution, obtained by treating current values of other unknowns ( $\gamma, \beta_1, \dots, \beta_k$ ) as true values.

In greater detail, our Step (1) is similar to the approach used by Lipkovich (2002). A unit is selected at random, which may be reassigned to a new class, depending on the outcome of the acceptance step. In case of more than two classes, the proposed new class is also selected at random, with equal probability of selecting each of  $k - 1$  classes.

Assuming that current and proposed classifications have equal prior probability, a proposal to reassign  $\text{GCU}_i$  from class  $j$  to class  $l$  is accepted with probability

$$\min \left\{ 1, \frac{f_N(\mathbf{y}_i; \mathbf{X}_i \boldsymbol{\beta}'_l, \sigma'^2 \mathbf{I})}{f_N(\mathbf{y}_i; \mathbf{X}_i \boldsymbol{\beta}'_j, \sigma'^2 \mathbf{I})} \right\},$$

where  $f_N(\mathbf{y}; \mathbf{m}, \mathbf{V})$  denotes a Gaussian density with mean vector  $\mathbf{m}$  and covariance matrix  $\mathbf{V}$ . (Here primes indicate variables that are subject to updating in an iterative scheme.)

If units were considered in strictly random order, the number of iterations between consecutive visits to a given unit could be large. Our scheme for randomly selecting units in Step (1) can be described as random in cycles of length  $n$ , with each unit visited once per cycle. At the start of the simulation we generate a random permutation of the indices  $1, \dots, n$  which gives the order that units are considered initially. The algorithm advances to the next unit in the current cycle each time the steps of the algorithm are executed, whether or not the constraints are met. At the end of the cycle, a second random permutation of unit indices is generated, which gives the order for visiting units in the next cycle, and so on.

To update the regression coefficient  $\boldsymbol{\beta}_j$ , with the classification fixed, we form vector  $\mathbf{y}_j$  from the observed responses for  $C_j$ , and a design matrix  $\mathbf{X}_j$  with corresponding values of regressors. Thus  $\mathbf{y}'_j{}^T = (\mathbf{y}'_{j1}{}^T, \dots, \mathbf{y}'_{j2}{}^T, \dots)$  where  $\mathbf{y}'_{j1}, \mathbf{y}'_{j2}, \dots$  are response vectors for GCU in  $C_j$  according to the current classification, and  $\mathbf{X}'_j{}^T = (\mathbf{X}'_{j1}{}^T | \dots | \mathbf{X}'_{j2}{}^T | \dots)$ . Fixing the classification, we update  $\boldsymbol{\beta}_j$  by drawing from a multivariate normal distribution with mean vector  $(\mathbf{X}'_j{}^T \mathbf{X}'_j)^{-1} \mathbf{X}'_j{}^T \mathbf{y}'_j$  and covariance matrix  $\sigma'^2 (\mathbf{X}'_j{}^T \mathbf{X}'_j)^{-1}$ . Fixing the classification and the regression coefficients we update  $\sigma^2$  using

$$\sigma^{2'} = \frac{1}{\chi_m^2} \sum_{i=1}^n (\mathbf{y}_i - \mathbf{X}_i \boldsymbol{\beta}'_{\gamma'_i})^T (\mathbf{y}_i - \mathbf{X}_i \boldsymbol{\beta}'_{\gamma'_i})$$

where  $\chi_m^2$  denotes a random draw from a chi-square distribution with  $m$  degrees of freedom.

To implement prior constraints -- on cluster sizes or distribution parameters -- we repeat the updating procedure until a set of unknowns is obtained that satisfies all constraints. In cases where we have had problems with our algorithm, the most common difficulty is that too many attempts have been required in order to satisfy all of our constraints. We may call this problem that of “inefficiency in implementing constraints.”

As described in greater detail in the next section, we use multiple independent chains to search for local optima and test for convergence.

### 3.1.3 Summarizing the Posterior Sample

Multiple independent chains may be generated, and pooled to form the posterior sample. We summarize the sample to display the evidence regarding the true classification and regression parameters. We emphasize summarization procedures described in Chapter 2, which do not depend on the labeling of classifications in the posterior sample.

To detect any tendency for a chain to become trapped in a part of the model space with low likelihood (*e.g.*, Celeux *et al.*, 2000, Ter Braak *et al.*, 2002), the logarithm of the likelihood

$$L(\boldsymbol{\beta}'_1, \dots, \boldsymbol{\beta}'_k, \sigma'^2; \mathbf{y}_1, \dots, \mathbf{y}_n) = \prod_{j=1}^k \prod_{i \in C_j} f_N(\mathbf{y}_i; \mathbf{X}_i \boldsymbol{\beta}'_j, \sigma'^2 \mathbf{I})$$

is plotted against position in a chain. Diagnosis of convergence is based on graphical comparison of multiple chains, using the scale factor of Gelman and Rubin (1992), as implemented in coda (Plummer *et al.*, 2005). Values of the scale factor for iterations 1-50, 1-60, 1-70, and so on, are plotted against the values 50, 60, and so on. We apply the procedure after thinning each chain, and deleting a burn-in from each.

Following the procedure described in Chapter 2, we emphasize summarization procedures based on the  $n \times n$  matrix **ECCP**. The value in the  $i$ th row and  $j$ th column,  $ECCP_{ij}$  say, is the sample fraction with  $GCU_i$  and  $GCU_j$  falling in the same cluster (also see Monti *et al.* 2003). Such a sample fraction estimates the posterior probability for an interesting event, and as such has a natural role in a Bayesian approach.

The matrix can be treated as a matrix of similarities and evaluated using methods such as cluster analysis. Examples of our graphical approach are Figures 13 and following, which we discuss in connection with analysis of specific data. Particular graphs that we find helpful are a dendrogram – we provisionally use average linkage -- and an ordination plot based on the method of principal coordinates (Gower, 1966). For the ordination plot **ECCP** is transformed to a non-negative definite matrix, and plotting positions for units are obtained from the eigenvectors of the transformed matrix.

We also use an approach based on the **ECCP** to compute clusters from simulation output. The dendrogram based on output of a  $k$ -class model is “cut” to identify  $k$  clusters. Graphically, this operation can be understood by drawing a line across a dendrogram at a critical dissimilarity value  $d^*$ , chosen such that a partition into  $k$  groups can be identified, from clusters joined with dissimilarity values no greater than  $d^*$ .

Some special graphs provided are relevant for determining the most appropriate number of classes. A common practice is to plot some measure of model quality against the number of classes assumed. It is common to look for a plateau in the functional relationship. Some measures of model quality tend to increase with an increasing number of classes, although the results with more classes may not be more reliable. We provide two plots of this type, based on different measures of model quality. For each measure, our present justification is informal.

First, we plot the median log-likelihood against  $k$ . This is roughly motivated by approximation of integrated likelihoods from MCMC samples, using the harmonic mean likelihood (Kass and Raftery, 2002), which is considered unstable (Kass and Raftery,



2002; Robert, 2001). Our second plot is based on our matrix **ECCP**. For a measure of uncertainty in the affinity of the  $i$ th and  $j$ th units we compute the quantity

$$U_{ij} = \text{ECCP}_{ij} (1 - \text{ECCP}_{ij}).$$

This quantity is averaged over  $n(n - 1) / 2$  pairs of units, and the average plotted against  $k$ .

### 3.2 AN APPLICATION TO DATA FROM MARYLAND WITH CLASS SPLINE REGRESSIONS

A promising approach for statistical modeling of ecological quality is to relate indices of ecological quality to watershed land cover fractions (King *et al.*, 2005). In this section we relate a multimetric measure of ecological quality to two land use variables, using data from Maryland streams. To provide an initial dataset with a good likelihood of interpretable regional variation, we restrict the analysis to data from two physiographic regions. The following analyses are presented.

- (1) Scatterplots and smoothing based on combined data, ignoring any regional variation.
- (2) Comparisons of regression surfaces between physiographic regions.
- (3) Model-based clustering with additive  $B$ -spline regressions.

Our simultaneous modeling of multiple land use categories is designed to extend previous analyses, which relied on bivariate scatterplots (Morgan and Cushman, 2005), by allowing a multidimensional regression surface.

#### 3.2.1 Data

The data analyzed were collected in 1995-1996 for the Maryland Biological Stream Survey (MBSS, Mercurio *et al.*, 1999), at 586 monitoring stations on non-tidal streams of order 4 or lower. Data collection was based on 2-stage sampling in each year, where the sampling units were basins and stream segments within basins. Basins to be sampled in a

given year were chosen at random, subject to a requirement that each was sampled in one or more years of the study.

Figure 3-1 displays coordinates of monitoring stations used in our analysis. Selecting cases using a categorical PHYSIO available with the data, we have focused on stations from the North Coastal physiographic region (“N,” 204 stations included) and the Piedmont region (“P”, 382 stations included). We also display in Figure 3-1 coordinates for 17 combinations of river basin (12) and region (2), averaging over coordinates for individual stations. These combinations are used for GCU in our model-based clustering. Table 3A.1 (appendix) gives numbers of stations for each GCU. (Basin is encoded in the MBSS variable BASIN.)

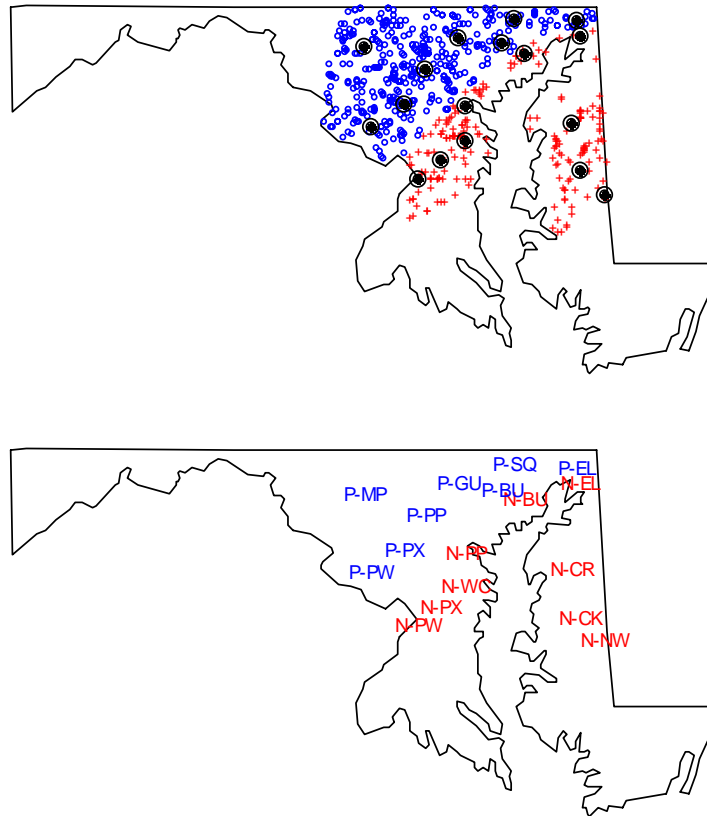
Our explanatory and response variables are

URBAN – percentage of watershed urbanized;  
AGRI – percentage of watershed in agricultural use; and  
BIBI – Benthic Index of Biotic Integrity (range 0-5).

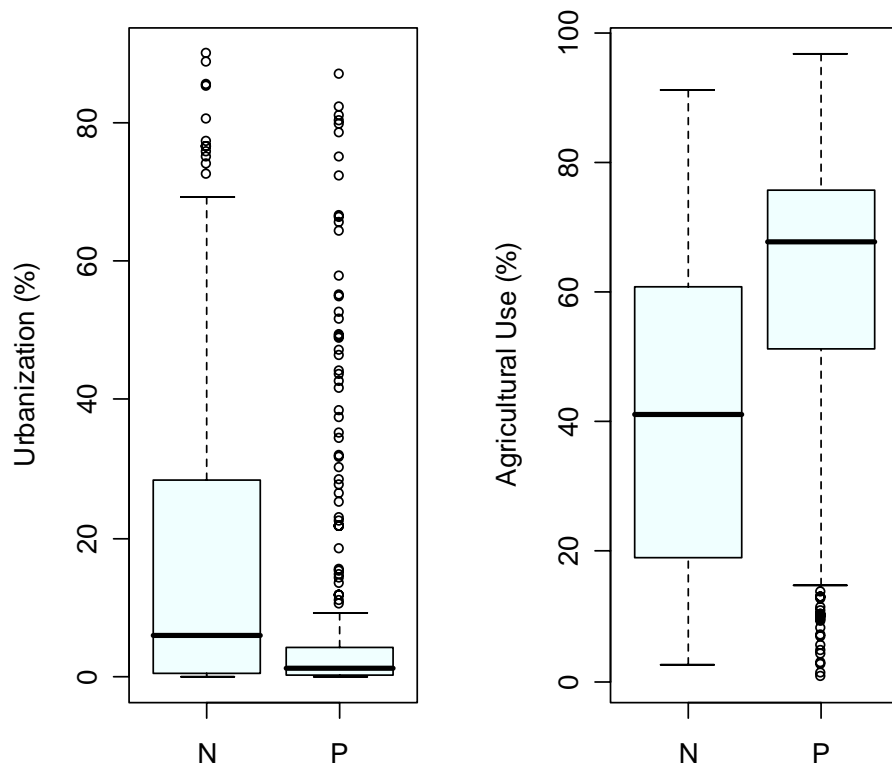
We take URBAN and AGRI to be explanatory variables and BIBI to be the response.

Of the possible land uses, urbanization and agriculture may be particularly associated with ecological stressors such as pesticides and other pollutants, sedimentation, and hydrological modification. Some additional land use categories are reported in the MBSS data. Most of land use not accounted for by our two categories is accounted for by forest.

Figure 3-2 compares distributions of the predictor variables for the two regions. Urbanization is more extensive in the North Coastal Region. However, each region displays a range of values, for each predictor.



**Figure 3-1** *Locations of Maryland monitoring stations and clustering units.*  
 The top figure gives locations of stations (“+” for stations in the North Coastal Region; “o” for the Piedmont). Large dots correspond to average coordinates for GCU. GCU labels are given in the lower figure, where the first digit encodes the region (N for North Coastal, P for Piedmont) and the last two digits the basin. For example, “N-PX” stands for the Patuxent River in the North Coastal Region, while “P-PX” stands for the Patuxent River in the Piedmont Region. Basins are identified in Table 3A-1 (Appendix).



**Figure 3-2** Distributions for two predictor variables, by physiographic region. (N = North Coastal Region; P = Piedmont Region).

### 3.2.2 Preliminary Analysis, with Comparison of Physiographic Regions

Preliminary analyses focused on the relationship of our response to the two predictors, separately for two physiographic regions (Piedmont, North Coastal), and for the combined data.

Generalized additive models (GAMs, Hastie and Tibshirani, 1990; Venables and Ripley 1994) were used to relate the response variable to the two predictors. According to the GAM approach as implemented for our context, the response is represented as a sum of smooth functions for two predictors,

$$f(\text{URBAN}, \text{AGRI}) = \text{constant} + f^{\text{urb}}(\text{URBAN}) + f^{\text{agr}}(\text{AGRI}). \quad (3.1)$$

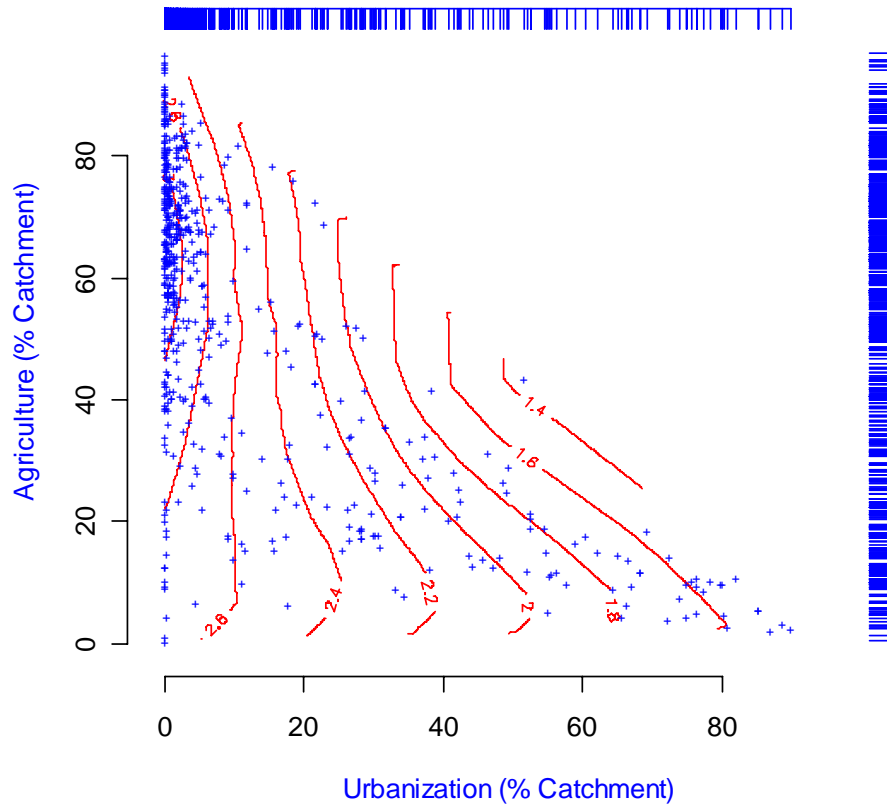
The fitted component functions,  $\hat{f}^{\text{urb}}$  and  $\hat{f}^{\text{agr}}$  say, can be viewed as summarizing effects of respective predictors. While various options are available for specification of the component functions, we have used the LOWESS specification for our preliminary analysis.

In addition, we use 2-dimensional LOWESS smoothing to generate surfaces such as our Figure 3-3. For these plots, which are not automated output of the GAM software, we evaluate a function on a grid of values of the predictors. We use two-dimensional linear interpolation for predictor combinations not present in the data, avoiding extrapolation outside the convex hull of combinations in the data.

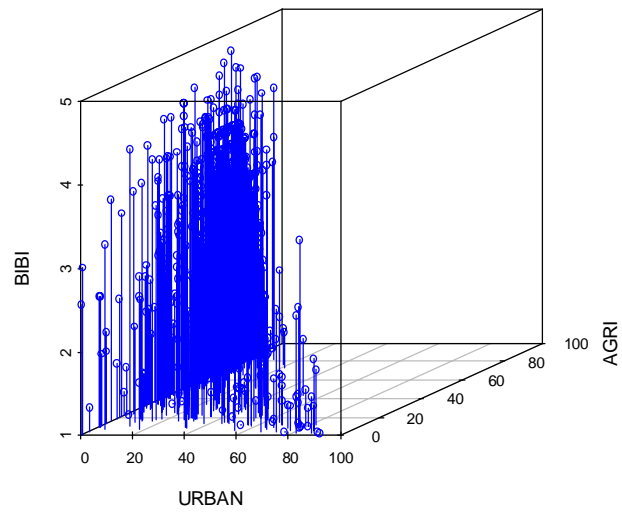
Results obtained with data combined for the two regions (Figures 3-3 to 3-6) suggest a roughly linear effect of urbanization, over most of the range of that variable. The role of agricultural use appears more complex, with an apparent optimum at low urbanization and intermediate agriculture.

However, efforts to interpret the roles of the predictors individually may be complicated by interactions. For our results from 2-dimensional smoothing, which allows interactions, the deviance has a value of 388. This value seems large relative to the deviance of 297 from our additive model. Corresponding approximate model degrees of freedom from the two approaches (see Ruppert *et al.*, 2003) are similar in value (576.6 and 576.5).

Fitted surfaces for the two regions are compared in Figures 3-7 and 3-8. (These figures were constructed by extracting using rows or columns from the grids such as were used for the 2-dimensional surface in Figure 3.3.) An obvious difference is higher ecological quality for the Piedmont region, particularly at low urbanization. For each region there is some indication of the optimum that we noted for the combined data.

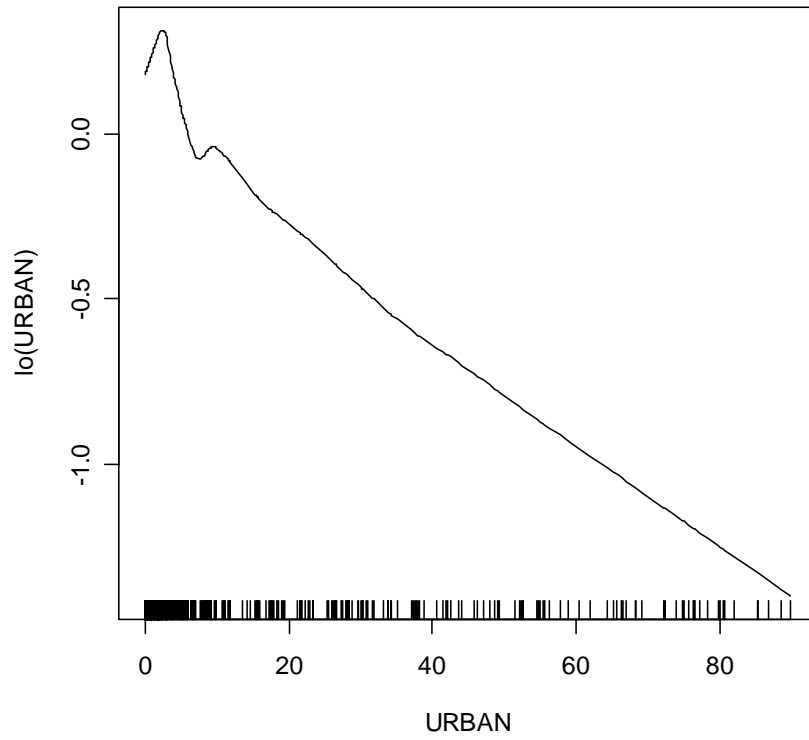


**Figure 3-3** 2-dimensional (non-additive) smooth relating BIBI to two land use variables, based on data combined for two regions. Plotted symbols represent predictor combinations with one or more observations of the response.

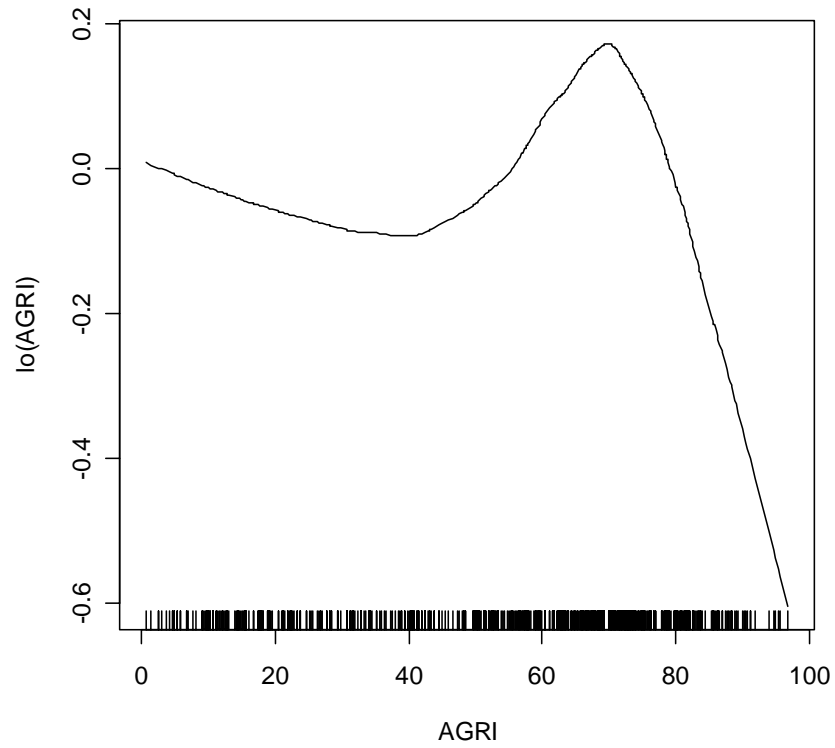


**Figure 3-4** 3-Dimensional scatterplot relating *BIBI* to two land use predictors.

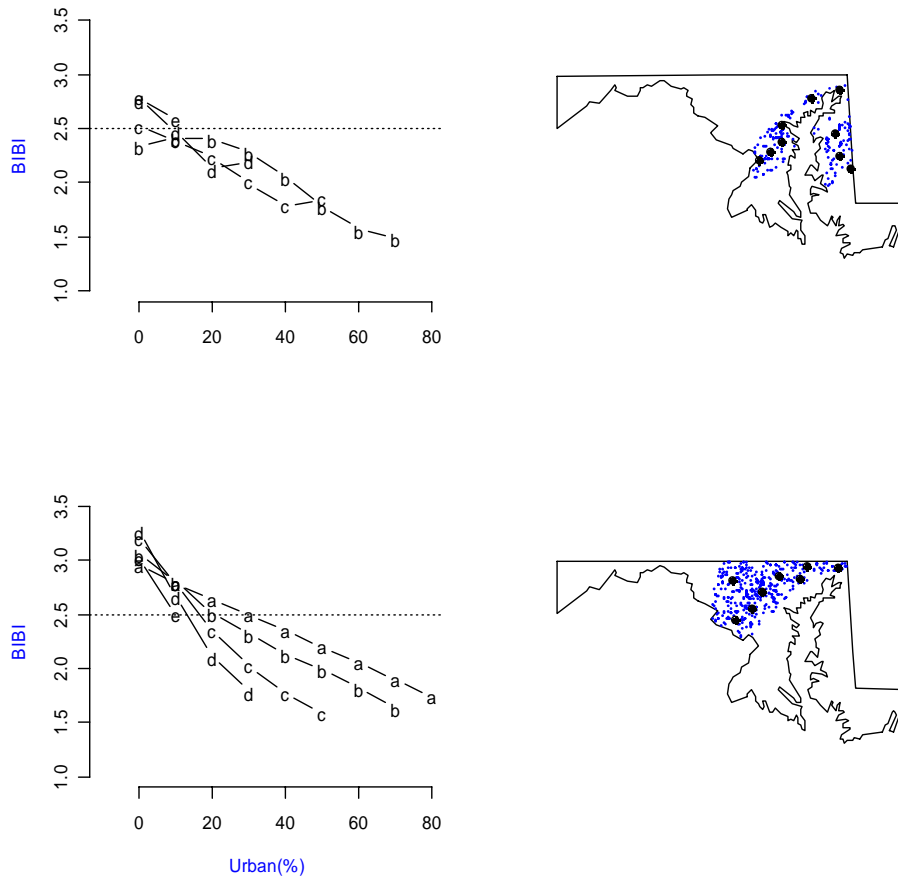




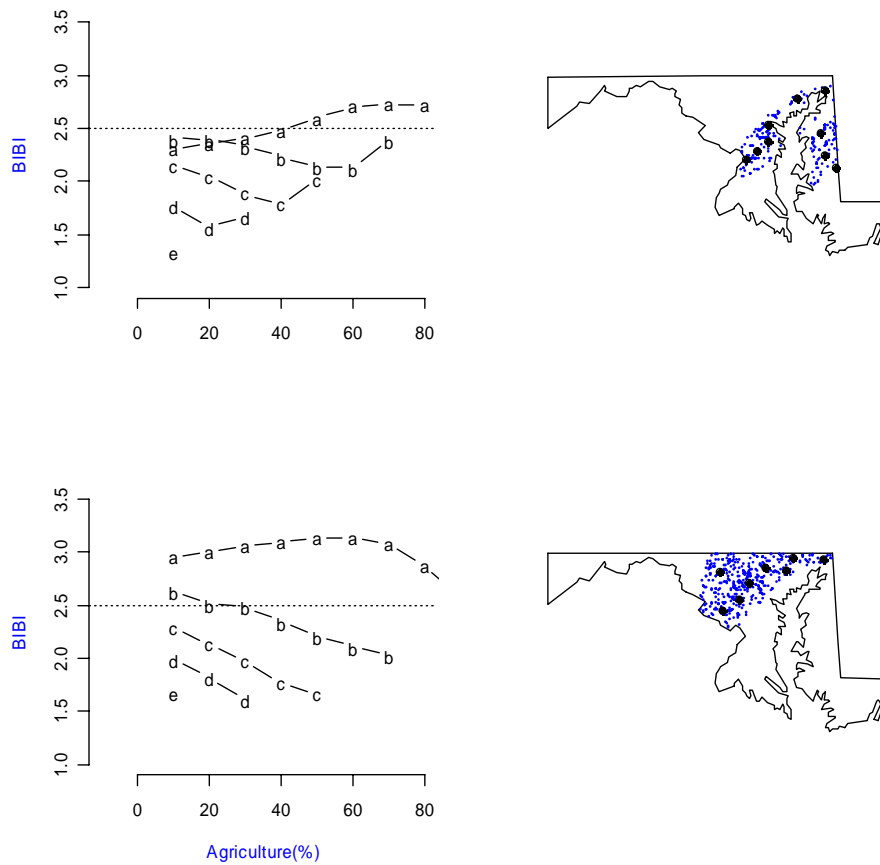
**Figure 3-5** *Urbanization component from an additive model fitted to the combined data.*



**Figure 3-6** *Agricultural component from an additive model fitted to the combined data.*



**Figure 3-7** Relationship of BIBI to urbanization for two physiographic regions. The two regions are North Coastal (upper graphs) and Piedmont (lower). Separate curves are displayed corresponding to agricultural usage a) 2.5%, b) 20%, c) 40%, d) 60%, and e) 80%. For each region we display the regression surface alongside geographic locations of stations.



**Figure 3-8** Relationship of BIBI to agricultural usage for two physiographic regions. The two regions are North Coastal (upper graphs) and Piedmont (lower). Separate curves are displayed corresponding to urbanization a) 2.5%, b) 20%, c) 40%, d) 60%, and e) 80%.

### 3.2.3 Model-Based Clustering with Additive B-Splines

Our preliminary analysis supports the use of a class regression model incorporating nonlinear functions of predictors. We use an additive class regression model with the same form as (3.1), now interpreted as giving the expected response. However, to exploit the model-based clustering approach based on class linear models, our components  $f^{\text{urb}}$  and  $f^{\text{agr}}$  are now taken to be cubic  $B$ -splines, as used in some previous clustering efforts (*e.g.*, Henderson, 2006; Qin and Self, 2006). For a given classification, cluster design matrices  $\mathbf{X}_1, \dots, \mathbf{X}_k$  each have nine columns, one with all values equal to 1, and four columns each for  $f^{\text{urb}}$  and  $f^{\text{agr}}$ . The component  $f^{\text{urb}}$  is thus a linear combination of four known  $B$ -spline basis vectors, based on a knot at 50%. Similarly, four spline functions are generated to represent the component  $f^{\text{agr}}$ , also with a knot at 50%.  $B$ -spline basis vectors were generated using the R library function `bs` (Venables and Ripley, 1998). To avoid redundant specification of an intercept in the bases for individual predictors, by appropriate use of a function argument, no column of 1's is included in the basis for a given predictor.

Our preliminary analyses suggests interactions among predictors. According to the usual approach for representing interactions in linear models, we include in the design matrix element-by-element products of columns corresponding to main effects. However, with 4 basis vectors for each predictor, the resulting model might be overparameterized without an effort to reduce the dimension of the model space. While, for the time, we have relied on an additive model for identification of clusters, we have applied 2-dimensional smoothing for clusters identified.

The method was implemented with a requirement of 20 or more monitoring stations per cluster, and an expected response in the prior range  $[0, 5]$  of the response variable. For a given number of classes, we generated 10 independent chains, each of length 20000, reduced to 10000 by saving the results from even-numbered iterations. The first half of

each chain was deleted to allow for equilibration, so that the final sample size was 5000 for each chain.

We use the label-invariant approach of the previous chapter to identify clusters from the simulation output. For each cluster we use 2-dimensional smoothing to relate the response variable to the predictors. The smoothing procedure is that used in Section 3.2.2 to generate displays for the combined data and for individual physiographic regions. We do not extrapolate function values beyond the convex hull of predictor combinations present in a given cluster. (While our spline functions can be evaluated for any combination of predictors, function values extrapolated to more distant combinations may be unstable, considering the flexibility of spline functions.)

An alternative more in keeping with the Bayesian approach would be to estimate posterior means for function values. However, for the number of regression coefficients in the spline models, the sample of regression coefficients requires more than 20 megabytes when results are stored in an R workspace, for analyses of sizes that we perform routinely. Also, our use of 2-dimensional smoothing allows for interactions, not accounted for in our additive model used in clustering.

#### *3.2.4 Results*

Figures 3-9 to 3-11 display log-likelihoods and our diagnostic plot for convergence for models with different  $k$ . (For the latter plot, convergence is indicated when the scale factor approaches a value of 1.) The results do not suggest any complications in relying on the pooled sample, for the 2-class model. With more than 2 classes different chains converge to different parts of the model space with likelihood values tending to differ. However, Figure 3-9 suggests 5 particular chains with similar, high likelihood. We have conducted several analyses using only these 5 chains. Figure 3-11 suggests convergence based on comparison of these chains.

Figure 3-12 relates the two measures of model quality to the assumed number of classes. According to common practice in interpreting such graphs, they suggest a 3-class model.

The plot based on likelihood plateaus at about 3 classes. The plot based on ECCP suggests that with 3 classes uncertainty in clustering relationships is maximized, and variation among chains is minimized.

Figures 3-13 to 3-16 provide a graphical evaluation of clustering results for 2- to 4-class models. In our ordination plots, centroids are circled for clusters generated from the dendrograms. (Diameters of circles in these figures are arbitrary. In particular, the diameter of a circle is not selected to quantify uncertainty or variation.) If we note the locations of individual units relative to the group centroids, the figures provide a sense of the uncertainty in associations of particular units, given the meaning of distances (Chapter 2). With 3 or fewer assumed classes, the graphs suggest that units can be assigned to classes with reasonable confidence.

Table 3-1 displays estimates of the probability of membership for a given unit, for each class, based on 5 selected chains from the 3-class model. The 5 chains were combined after relabeling according to the approach of Chapter 2. Results are in good qualitative agreement with Figure 3-15, for the purpose of identifying the units with the greatest uncertainty in class membership.

In Figures 3-17 to 3-22, coordinates of stations assigned to a cluster are displayed alongside estimated class regression surfaces. In the 3-class solution, the 2 larger clusters are similar to those from the 2-class solution. The maps suggest that some clusters take the form of one or more “swaths” of nearby units. For the 2-class model 5 of 7 units in one cluster belong to the North Coastal region, while 4 of 10 units belong to the North Coastal region for the other cluster. It is of interest to compare these results to results based on predefined regions, and on the combined data.

### *3.2.5 Discussion*

We note that 5 of our 12 river basins are represented in both regions (Table 3-2, Appendix). There is no apparent tendency for GCU that represent the same basin to be assigned to the same cluster, or to different clusters.

Clusters found using the 2-class model are similar to the larger two clusters, found with the 3-class model, with respect to units included, and fitted surfaces (Figures 3-17 to 3-22). For 2- or 3-class results one cluster (K2B or K3C) has a relatively high fraction of Piedmont units. Another (K2A or K3A) seems to divide into 3 swaths. For the latter, many units belong to the North Coastal region, but the western-most Piedmont unit -- the Middle Potomac basin -- is also included. Most stations in that basin are in the Monocacy watershed, but some smaller Potomac tributaries are also represented.

It seems plausible for the Middle Potomac basin to be clustered with some coastal units. According to the USEPA Level IV ecoregions classification (Woods *et al.*, 1999), the basin corresponds roughly to the Region 64a, the Triassic Lowlands, one of two Northern Piedmont sub-regions. The region has typically low elevation relative to the remainder of the Piedmont.

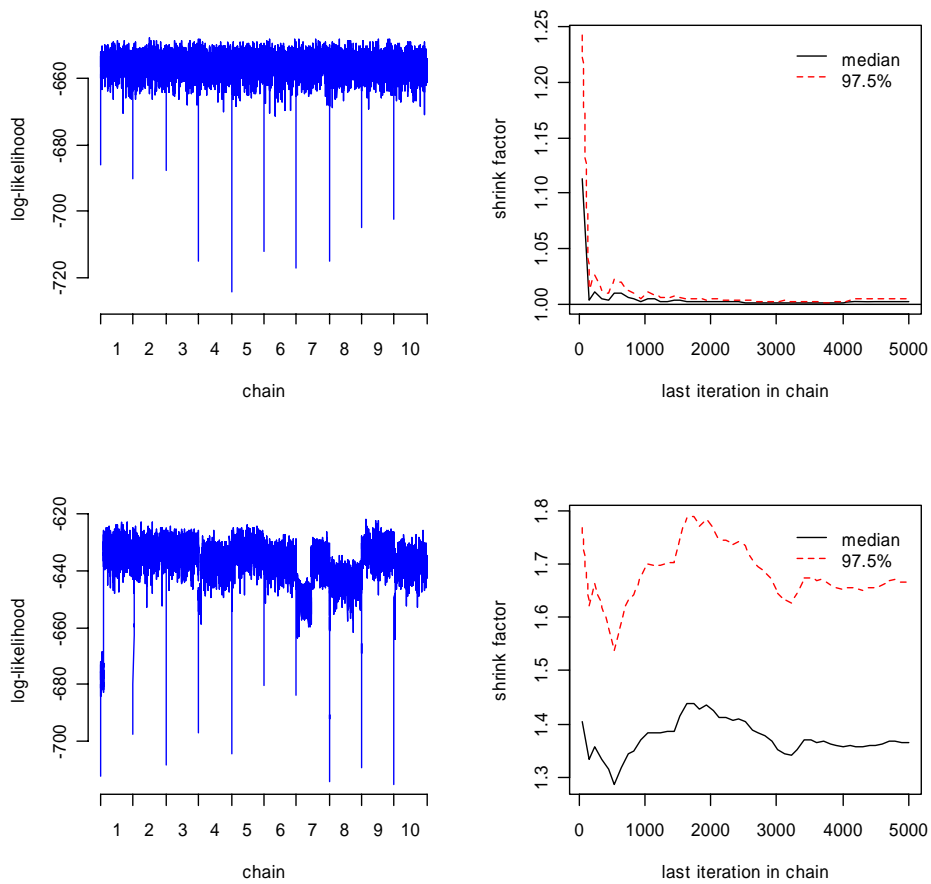
Our results divide the Northern Coastal Plain differently from the Level IV classification, which recognizes 3 Level IV categories. However, it is difficult to compare our results for North Coastal units to Level IV categories because -- unlike with the Middle Potomac basin -- most coastal plain basins include portions of multiple Level IV categories.

Some of our surfaces suggest an interaction between predictors, such that decrease in ecological quality with increase in agricultural use is steeper at high than at low urbanization. For the combined data, we have noted evidence of an actual optimum of ecological quality at intermediate agriculture and low urbanization (Figure 3.3). However, our 3-class results (Figure 3-22) suggest that the feature may be accounted for by a subset of units assigned to cluster K3B.

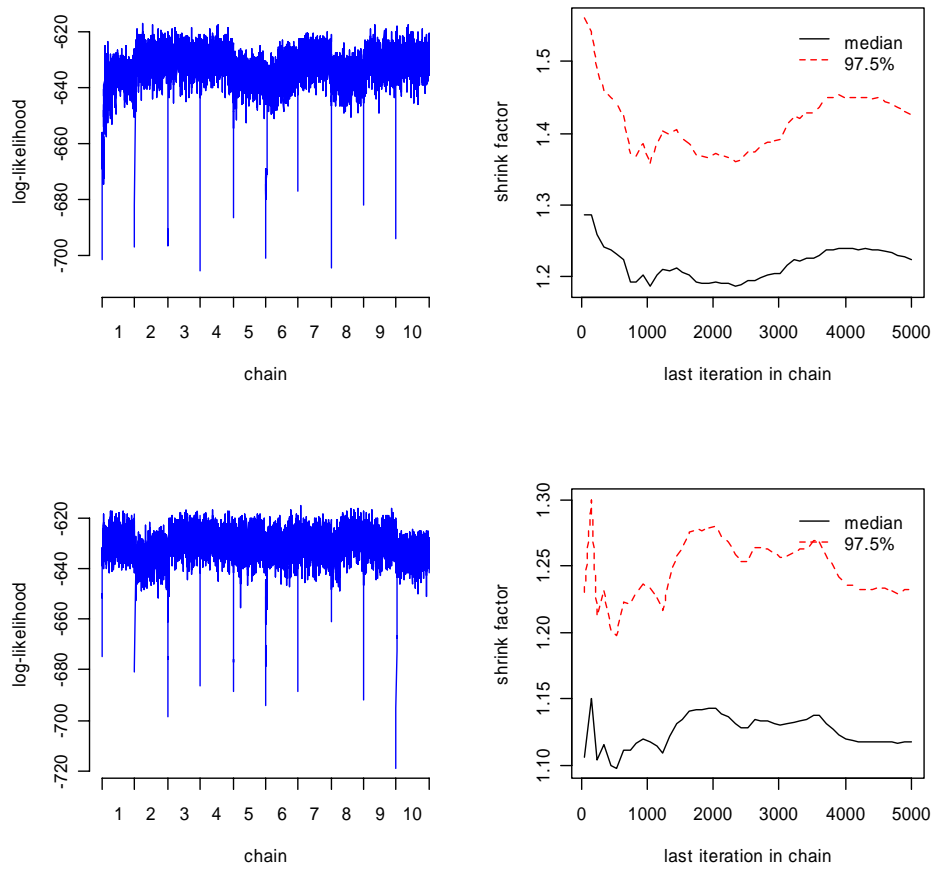


**Table 3-1** *Estimated probabilities of class membership for each unit for the Maryland data, based on selected chains from 3-class model. ('-' – estimated frequency is zero.)*

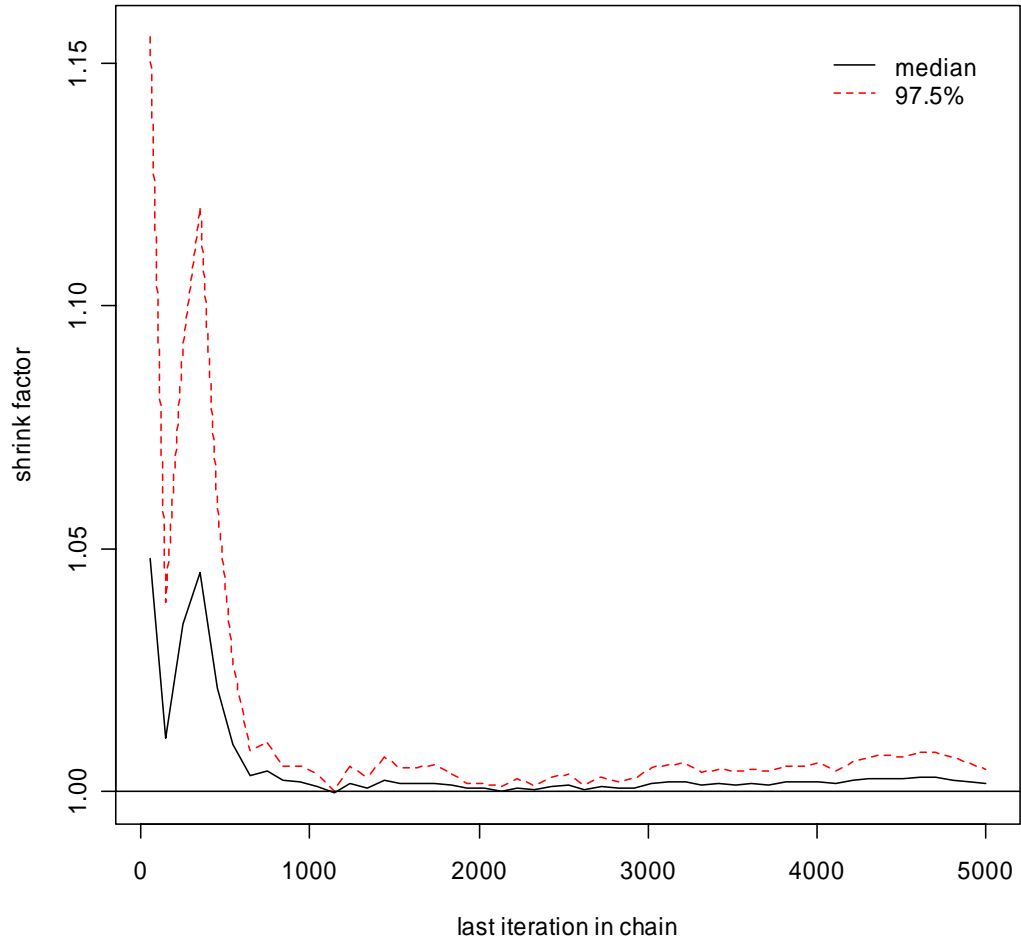
Unit	Assigned Class	Class Membership Probability, by Class		
		$C_1$	$C_2$	$C_3$
N-BU	1	0.999	-	0.001
N-CK	1	1	-	-
N-CR	2	-	1	-
N-EL	2	0.047	0.912	0.042
N-NW	1	0.769	0.150	0.081
N-PP	3	0.006	-	0.994
N-PW	3	0.003	-	0.997
N-PX	1	1	-	-
N-WC	2	-	1	-
P-BU	3	0.227	0.002	0.771
P-EL	1	0.931	0.057	0.012
P-GU	3	-	-	1
P-MP	1	1	-	-
P-PP	3	-	-	1
P-PW	3	-	-	1
P-PX	3	-	-	1
P-SQ	3	-	0.001	0.999



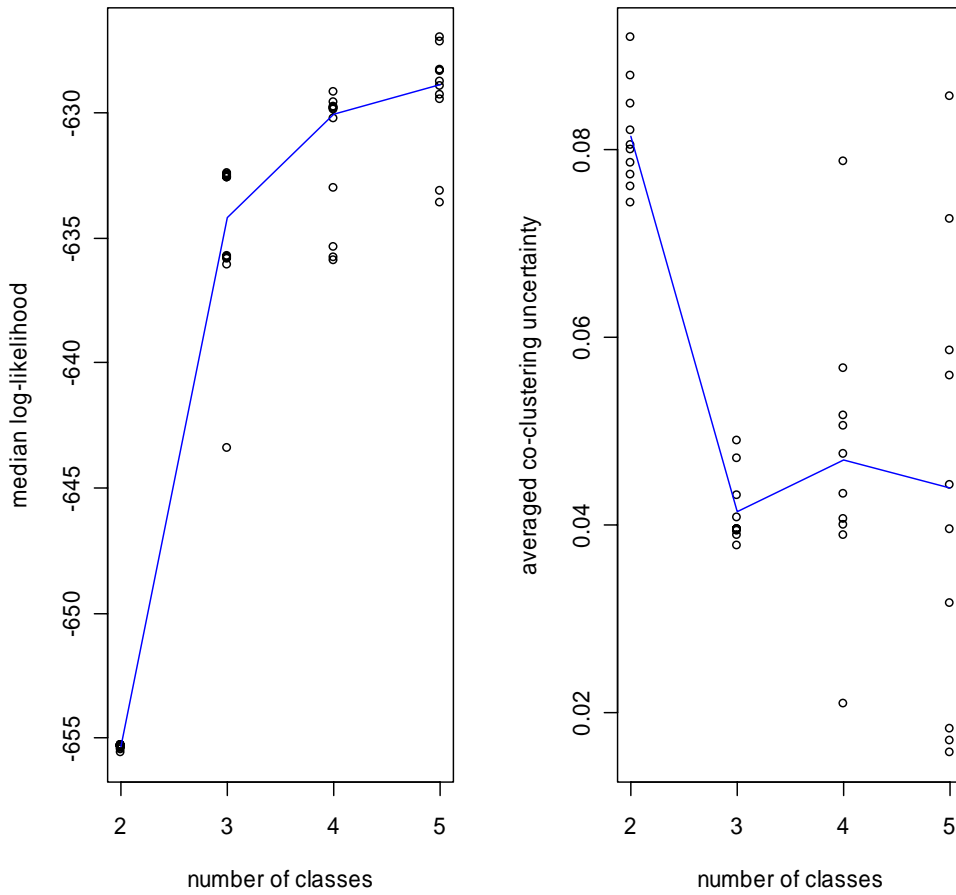
**Figure 3-9** *Ten Metropolis chains each for 2- and 3-class models.* The upper graphs are based on the 2-class model, the lower graphs on the 3-class model. For each model we display the log-likelihood on the left and a diagnostic plot for convergence on the right.



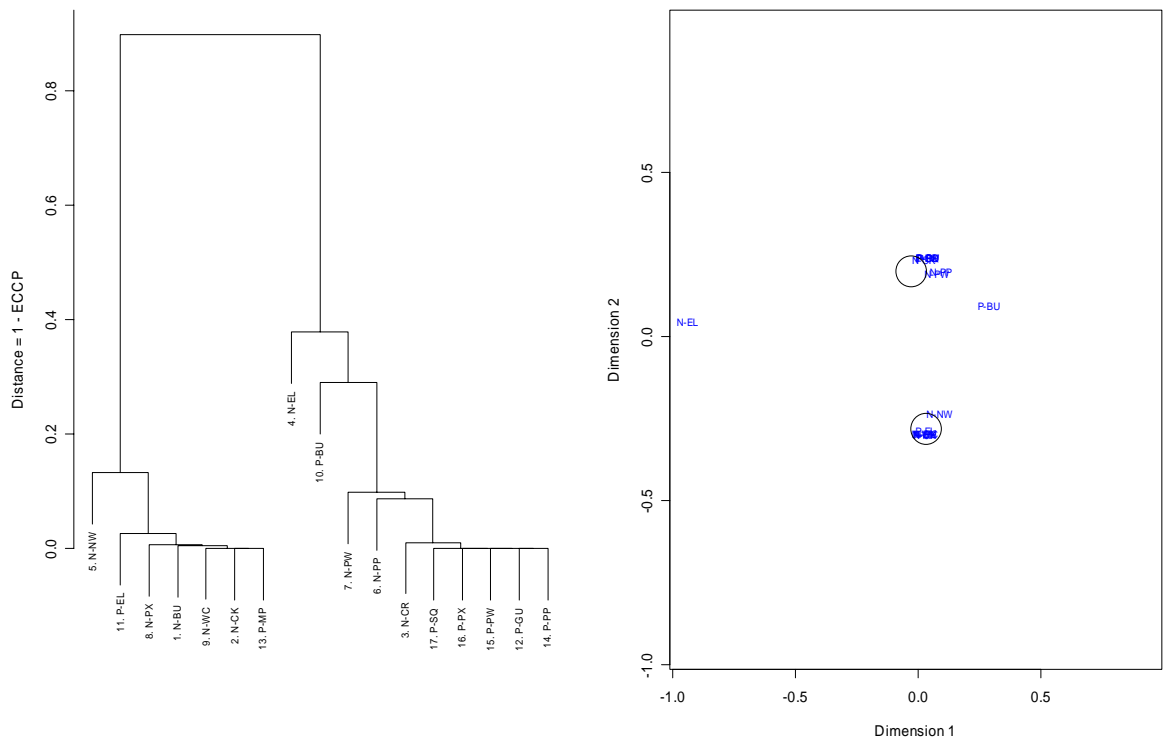
**Figure 3-10** *Ten Metropolis chains each, for 4- and 5-class models.*  
 (Top: 4-class model; bottom: 5-class model; left: log-likelihood;  
 right: convergence diagnostic plot.)



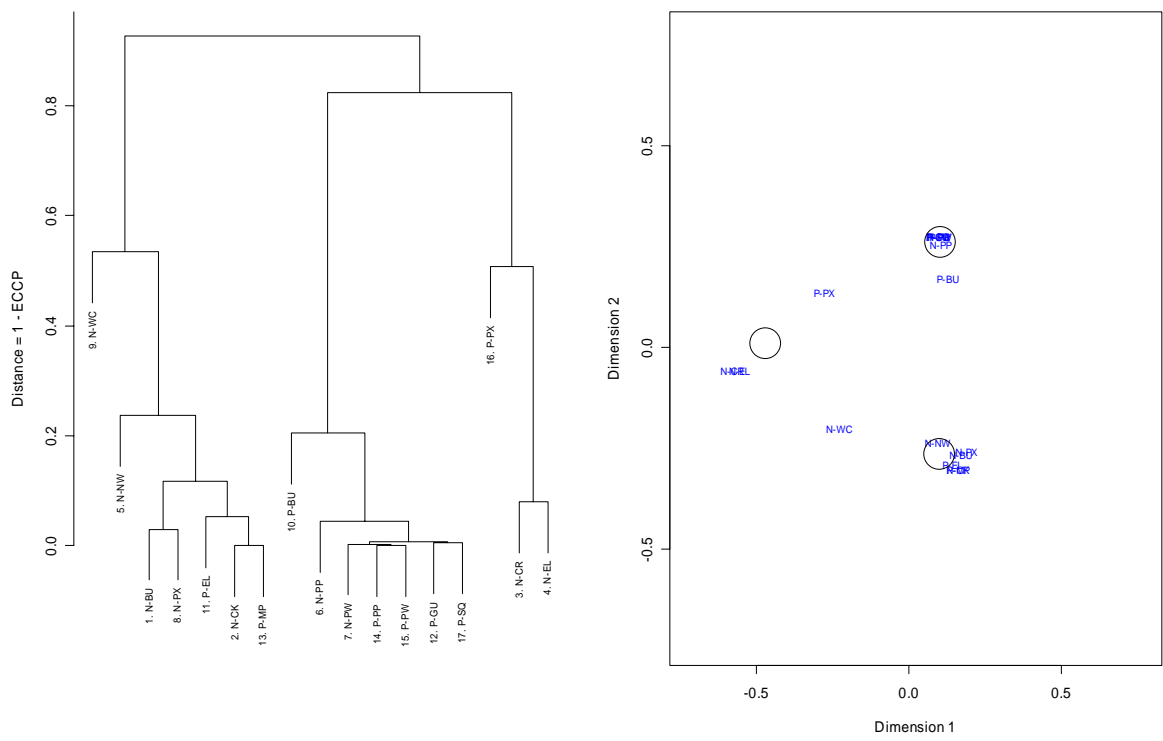
**Figure 3-11** *Convergence diagnostic plot for 5 selected chains from 3-class solution.* The graph is based on the chains numbered 1, 2, 3, 5 and 9 in Figure 3-10.



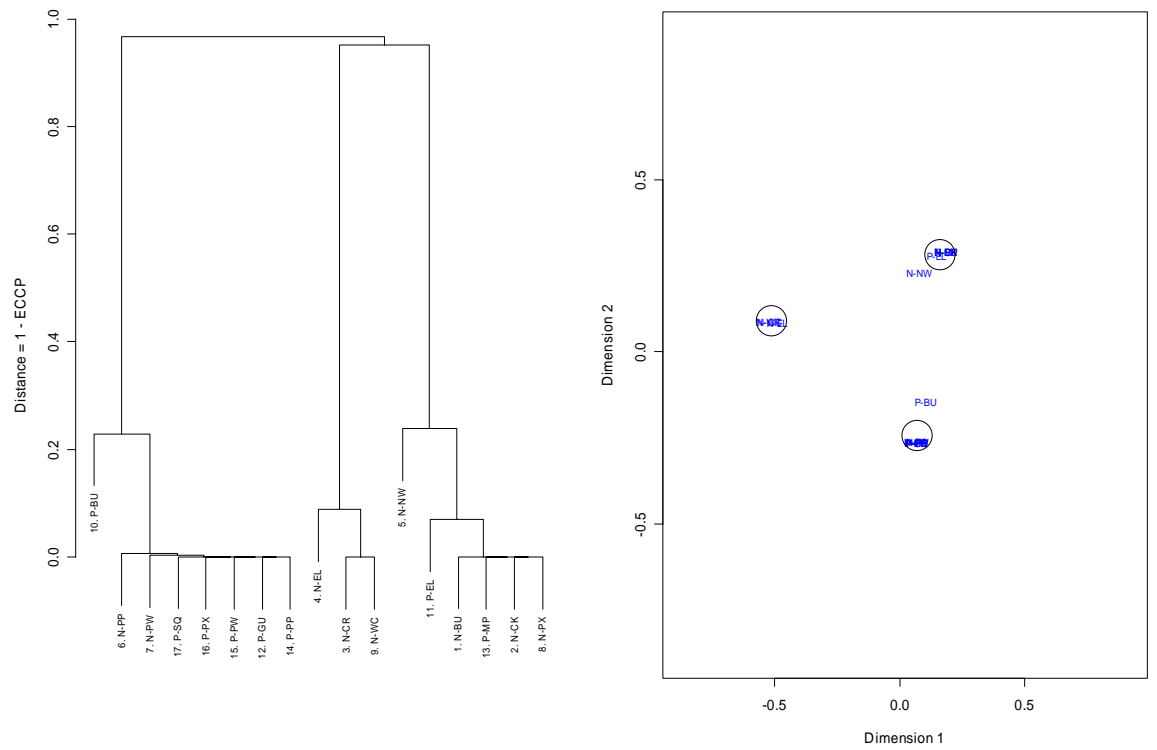
**Figure 3-12** *Measures of model quality related to the number of classes assumed.* Each point represents a single chain. For each plot, line segments connect the medians (left) or means (right) for 10 chains.



**Figure 3-13** *Dendrogram and ordination plot based on ECCP from 2-class model. (In the ordination plot, circles mark the centroids of clusters identified from the dendrogram. The circle diameter is arbitrary.)*

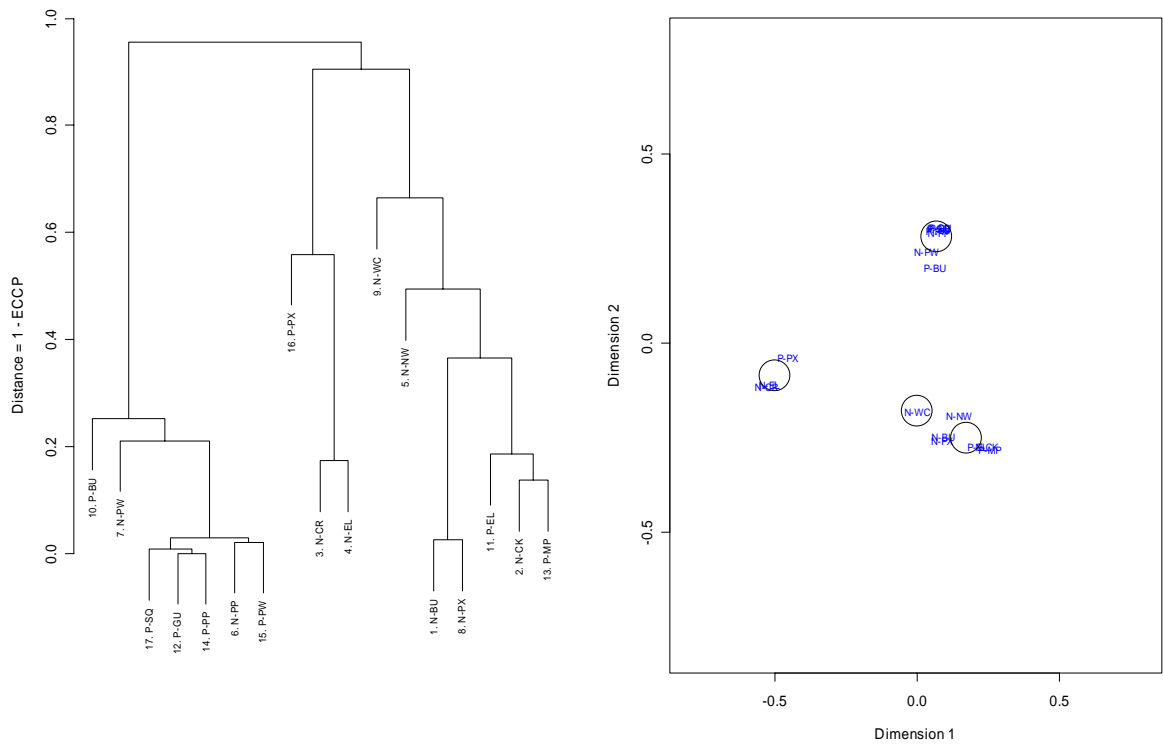


**Figure 3-14** Dendrogram and ordination plot for 3-Class model (see also Figure 3-15).

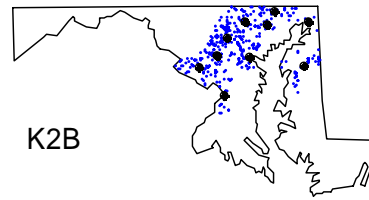
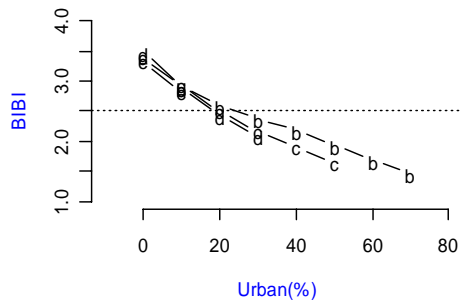
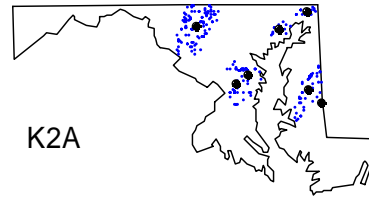
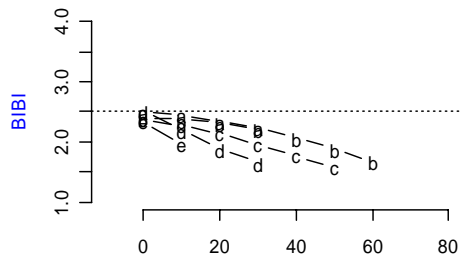


**Figure 3-15** Dendrogram and ordination plot based on 5 selected chains, for 3-class model.

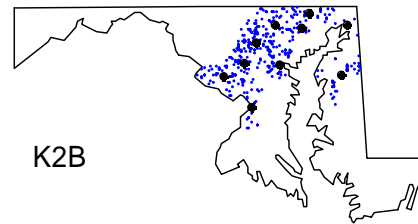
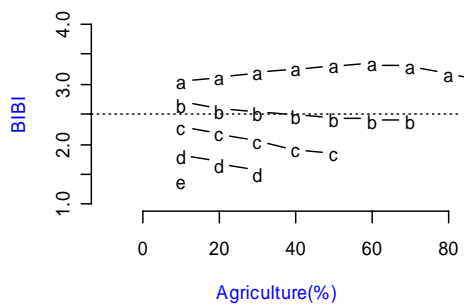
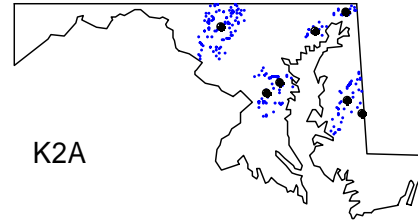
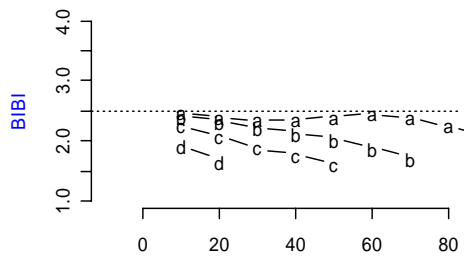




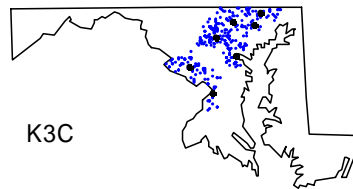
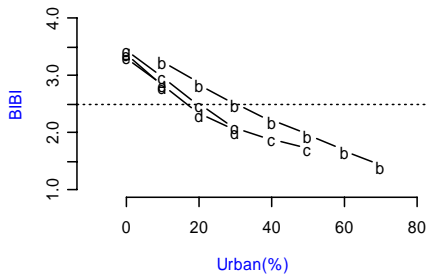
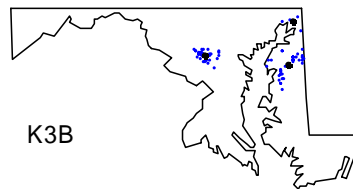
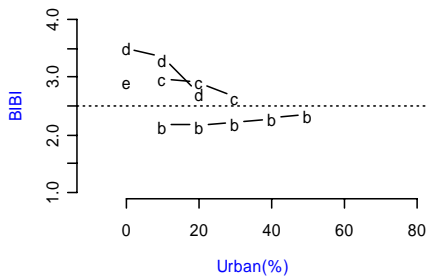
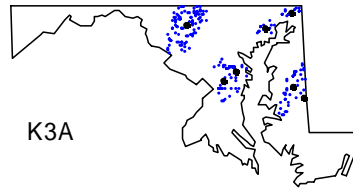
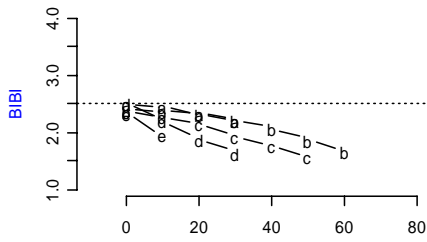
**Figure 3-16** Dendrogram and ordination plot for 4-class model.



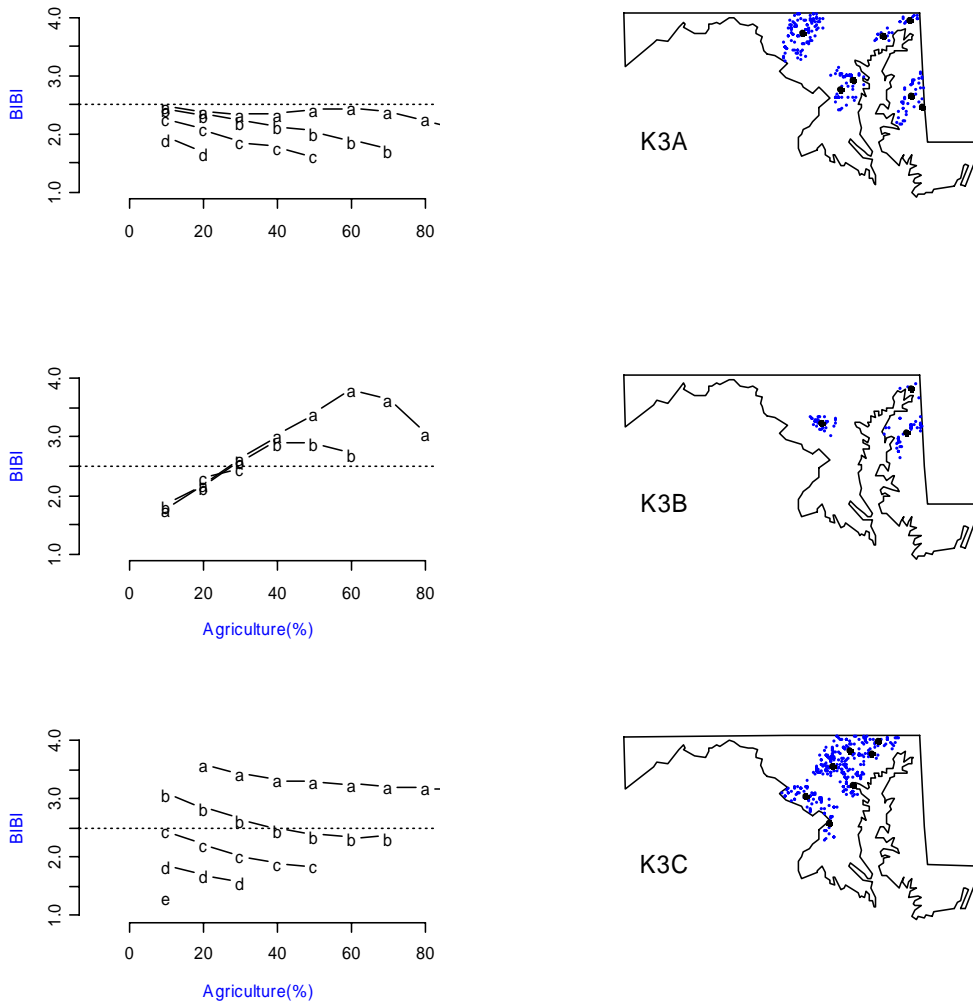
**Figure 3-17** Relationship of BIBI to urbanization, based on 2-class model. Separate curves relate BIBI to urbanization, fixing agricultural use at a) 2.5%, b) 20%, c) 40%, d) 60%, and e) 80%.



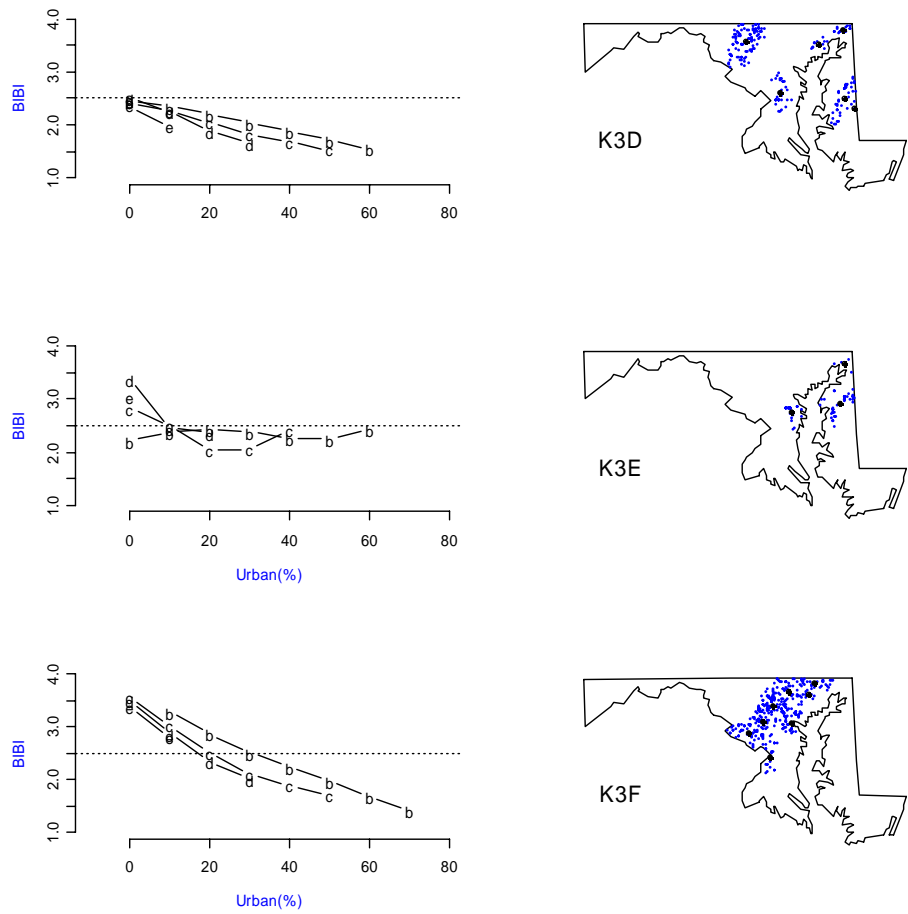
**Figure 3-18** Relationship of BIBI to agricultural use, based on 2-class model. Separate curves are displayed corresponding to urbanization a) 2.5%, b) 20%, c) 40%, d) 60%, and e) 80%.



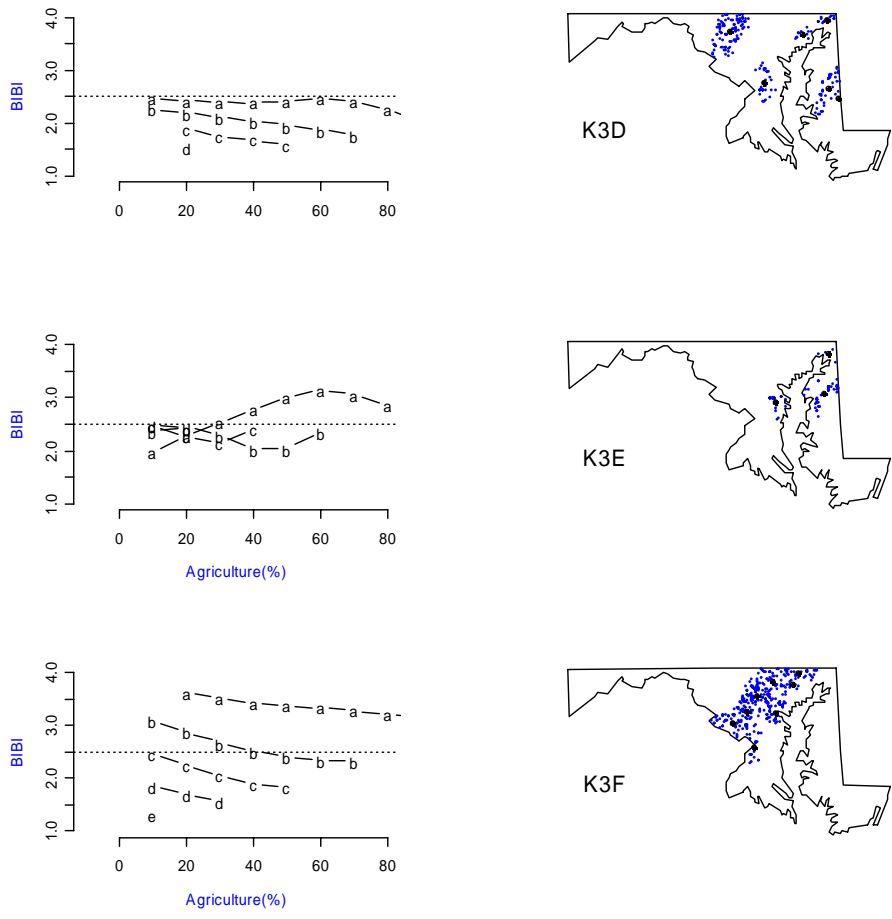
**Figure 3-19** Relationship of BIBI to urbanization, based on 3-class model. Separate curves are displayed corresponding to agricultural use a) 2.5%, b) 20%, c) 40%, d) 60%, and e) 80%.



**Figure 3-20** Relationship of BIBI to agricultural use, based on 3-class model. Separate curves are displayed corresponding to urbanization a) 2.5%, b) 20%, c) 40%, d) 60%, and e) 80%.



**Figure 3-21** Relationship of BIBI to urbanization, based on 5 selected chains from 3-class results. Separate curves are displayed corresponding to agricultural use a) 2.5%, b) 20%, c) 40%, d) 60%, and e) 80%.



**Figure 3-22** Relationship of BIBI to agricultural use, based on 5 selected chains from 3-class results. Separate curves are displayed corresponding to urbanization a) 2.5%, b) 20%, c) 40%, d) 60%, and e) 80%.

### 3.3 AN APPLICATION TO DATA FROM OHIO, WITH CLASS LINEAR MODELS

Our second example is based on data from streams and rivers in Ohio, USA. For these data we adopted a model with linear class regressions after observing, in preliminary analyses, and no indications of nonlinearity in the relationship to predictors (presumably because of high error).

#### 3.3.1 Data

The data are a subset of those assembled by Dyer *et al.* (1998, DWCSW) representing streams and rivers in Ohio, USA. The original data were compiled from multiple sources. We use data representing 303 stream segments, and representing 16 of the 23 river basins in Ohio identified by Dyer *et al.* (see chapter appendix). For purposes of our clustering approach, the segments will serve in the role of “monitoring stations,” although an observation for a segment may summarize results from multiple actual stations. Geographic coordinates of segments are mapped in Figure 3-23, along with averaged coordinates for basins, using an undetermined coordinate system.

Our response variable is the Index of Biotic Integrity (IBI), a measure of the health of fish communities. Our regressors are:

DO\_MED = site median dissolved oxygen concentration (mg/L);  
PH\_MED = site median pH;  
ZNTOT\_MED = site median total zinc ( $\mu\text{g/L}$ ); and  
QHEI = Qualitative Habitat Evaluation Index.

We will usually refer to the regressors simply as DO, pH, zinc, and QHEI. From the data available we used the segments that had complete data for IBI and for four regressors. The original values for IBI and the QHEI are single values per segment. For other variables, Dyer *et al.* computed the median of available values for a segment.



IBI and QHEI are “multimetric indices” computed by summing the values of contributing “metrics” (references in Dyer *et al.*). Each contributing metric has a restricted range of values, resulting in a range of 0-60 for IBI.

Zinc is toxic to aquatic organisms at high concentrations. For pH, low values represent acidification, which has the effect of increasing the availability of toxic metals (such as zinc) to aquatic organisms. pH is negatively correlated with zinc in our data. Low DO may exclude some aquatic species and is often a consequence of eutrophication, for example resulting from use of fertilizers.

Zinc and oxygen concentrations were transformed to logarithms. (pH is already defined as the logarithm of the concentration of hydrogen ions.) We centered and scaled each regressor, to facilitate comparison of the estimated coefficients. Centering and scaling was carried out after logarithmic transformation where applicable.

Preliminary analyses included scatterplots and a multiple linear regression of IBI on the four predictors. For the model-based clustering, we placed sign constraints on regression coefficients. The multiple regression results are consistent with the sign constraints that we imposed: For each coefficient, the sign of the estimate from our regression agrees with the constraint, and the  $t$  statistic exceeds 3 in absolute value.

### *3.3.2 Model-Based Clustering with Linear Class Models*

The implementation of our approach for the Ohio data involves constraints on model unknowns, and a particular definition of clustering units. All regression coefficients except that for zinc were constrained to be non-negative. The coefficient for zinc was constrained to be less than or equal to zero. Additional constraints are a minimum count of 15 measurement stations per cluster (combining the counts for all basins belonging to a cluster). The analysis was performed using river basins as GCU.

We initially attempted a simulation with the number of chains, length per chain, and number of classes used for our analysis of the Maryland dataset. However, technical

difficulties were encountered with the 3-class model -- our problem of “inefficiency in implementing constraints” of Section 3.1.2. For the 3-class model we ran 5 chains of length 7500, thinned each chain 50%, and deleted the first half of each chain.

These results were used for analyses displayed in Figures 3.24-3.25. Analyses in subsequent figures, which involve class regression coefficients, are based on an additional chain with a length of 95000, after 50% thinning and deletion of a burn-in of length 5000. For the latter simulation, a plot of the likelihood was constructed but displayed no features of interest. The relabeling program generated no instances of apparent label switching.

### *3.3.3 Results*

Mixing appears rapid, based on the plots of the log-likelihood and the convergence diagnostic plots (Figure 3.24). Plots of our measures of model quality against number of classes (Figure 3.25) do not suggest that the likelihood has reached a plateau with 4 classes. The plot of averaged co-clustering uncertainty favors a 2-class model.

We suggest that it is useful to focus on results from the 2-class model. Clustering results for models with 2-4 classes are depicted in Figures 3.26 to 3.28. A clear pattern of clustering is observed with 2 classes. With 3-4 classes, we encounter two large clusters, which are similar to clusters found with the 2-class model. Our classification based on the 2-class model is displayed in Table 3.2.

Figure 3-29 to 3-33 display spatial locations of units belonging to our two clusters, alongside estimated posterior distributions of class regression coefficients, which we depict using boxplots. The maps suggest that clusters are separated roughly from north to south. However, the north-south separation is apparent in the western part of the state, while in the eastern part of the state no clear separation is evident. Of 6 northeastern basins, 4 are assigned to our predominantly northern cluster (07-Asht, 13-Rock, 18-Maho, 19-Cuya) but two belong to our predominantly southern cluster (03-Gran and 15-Chag).

Our predominantly northern cluster has an intercept relatively closer to zero. Given our centering of the regressors, the intercept is approximately a mean of the response, at least for typical values of regressors, suggesting lower ecological quality in the northern cluster. That cluster also has a more negative zinc coefficient, suggesting relatively greater sensitivity to that chemical, and QHEI coefficient closer to zero, suggesting relatively less benefit associated with better habitat quality. The separation of posterior distributions between clusters is less clear for coefficients associated with other predictors.

#### *3.3.4 Discussion*

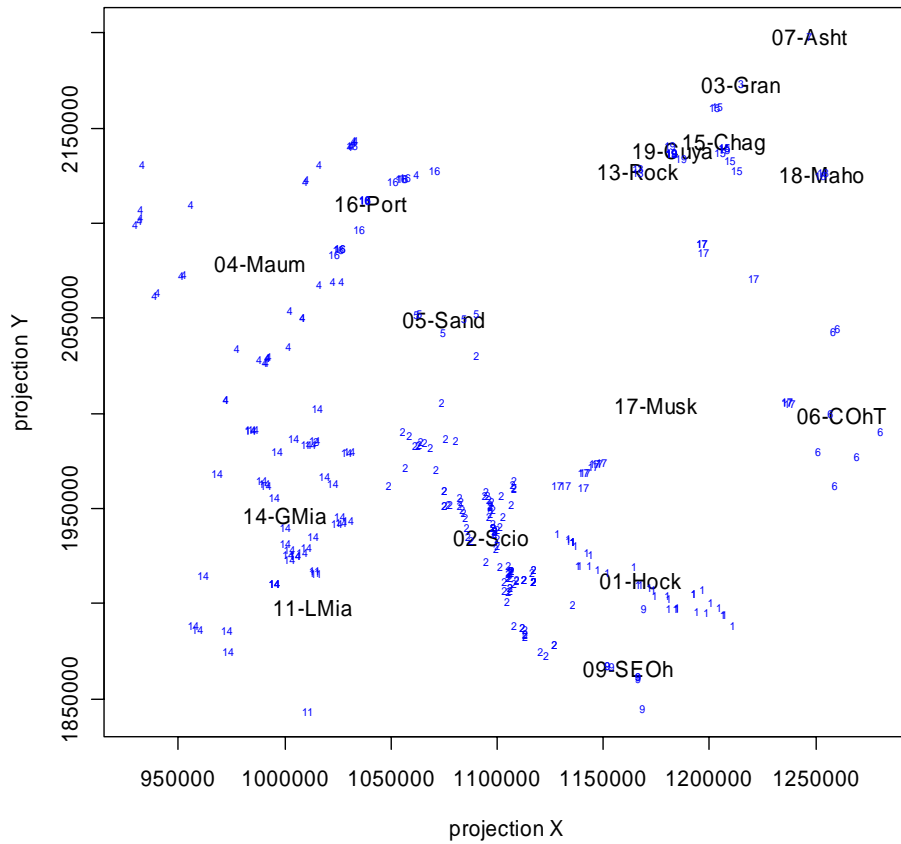
Our relatively northern cluster K2B belongs to an extensive area of lowlands adjacent to Lake Erie. Lowland streams can have low ecological quality according to some measures, perhaps sometimes from warmer water and lower oxygen pressure. More refined interpretations may need to account for additional variables. Suggestions from our results, regarding the roles of predictors, may be taken as hypotheses for future study.

Ohio includes parts of four USEPA Level III ecoregions. The ecoregion system recognizes a low plain adjacent to the Lake Erie -- the Huron and Erie Lake Plains ecoregion -- roughly in agreement with our results. However the correspondence between ecoregions and our clusters may be somewhat obscured because many basins include parts of more than one ecoregion.

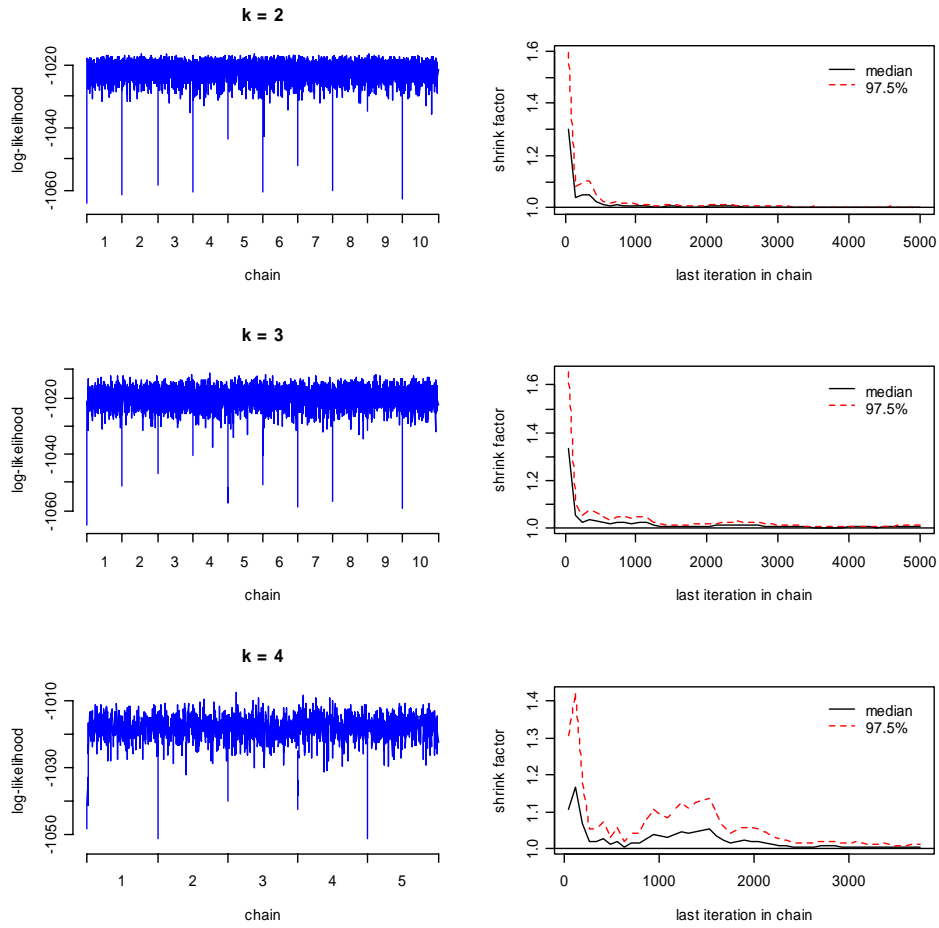
**Table 3-2** *Classification of Ohio basins based on 2-class model.*

<b>Cluster</b>	<b>Units Included</b>	<b>Probability for Assigned Class<sup>1</sup></b>
K2A	01-Hock	1
	02-Scio	1
	03-Gran	0.938
	06-COhT	1
	09-SEOh	1.00E-0
	11-LMia	0.928
	14-GMia	1
	15-Chag	0.751
	17-Musk	1.00E-0
K2B	04-Maum	1
	05-Sand	0.990
	07-Asht	0.607
	13-Rock	0.926
	16-Port	0.960
	18-Maho	0.719
	19-Cuya	1

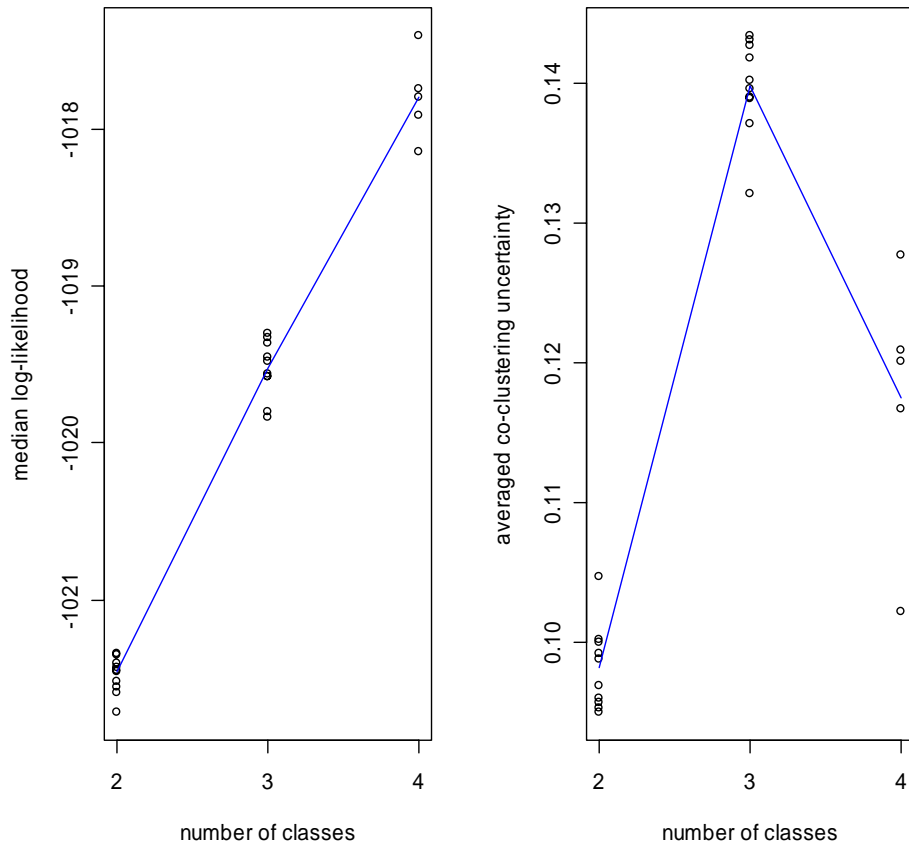
<sup>1</sup> Estimated posterior probability that unit belongs to K2A (for units we assigned to K2A) or that unit belongs to K2B (for our K2B units).



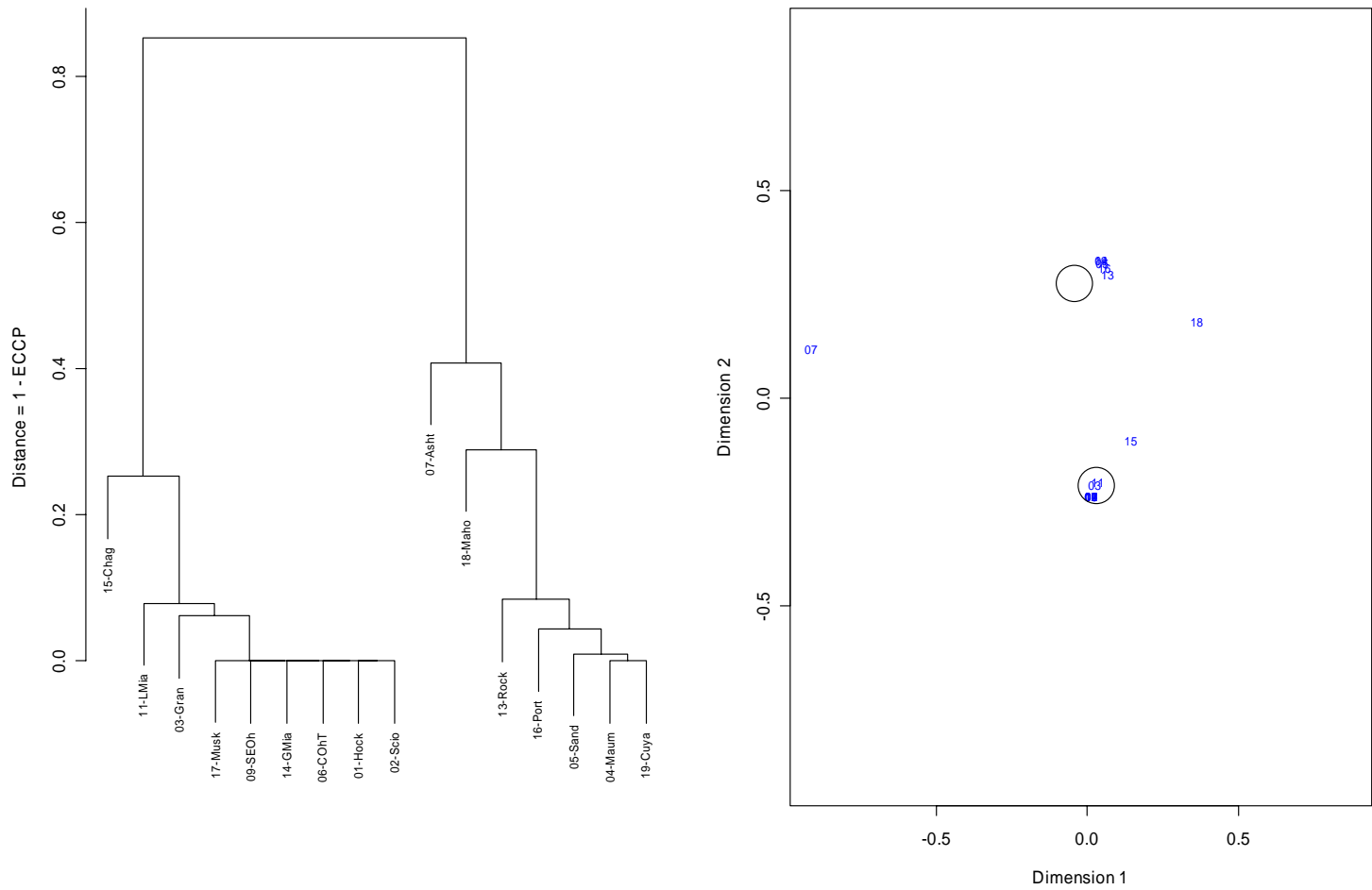
**Figure 3-23.** *Locations of Ohio stations and basins.* Basin labels (see chapter appendix) are plotted at averaged coordinates. The numbering of basins follows Dyer *et al.* (1998, see chapter appendix). (There are gaps in the numbering, for the data subset we have analyzed.)



**Figure 3-24** *Metropolis chains for models with 2-4 classes (Ohio dataset).* For each model we display the log-likelihood on the left and the diagnostic plot for convergence on the right.

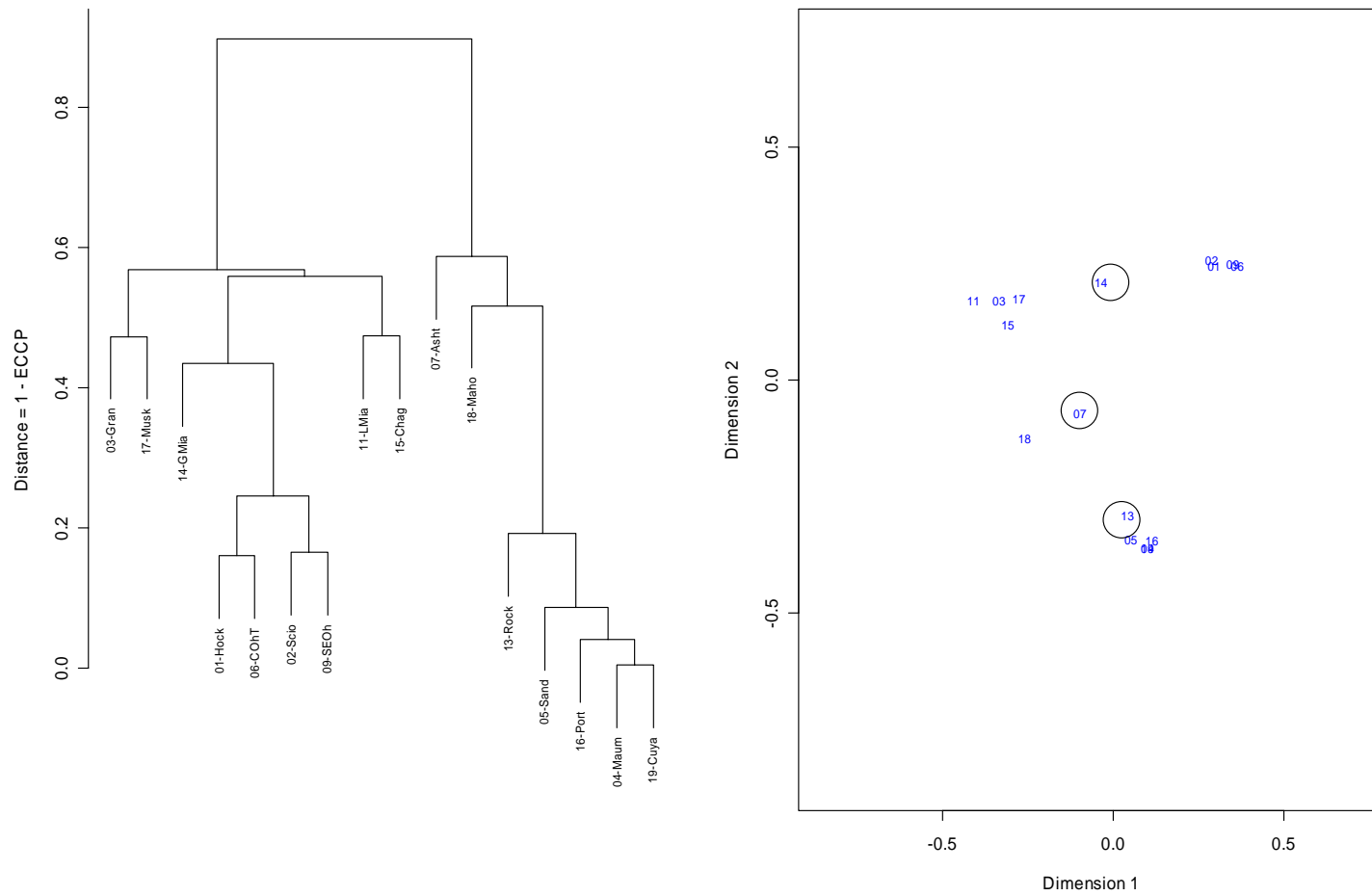


**Figure 3-25** Measures of model quality related to the number of classes assumed (Ohio dataset). Each point represents a single chain. The lines connect medians (left) or means (right).

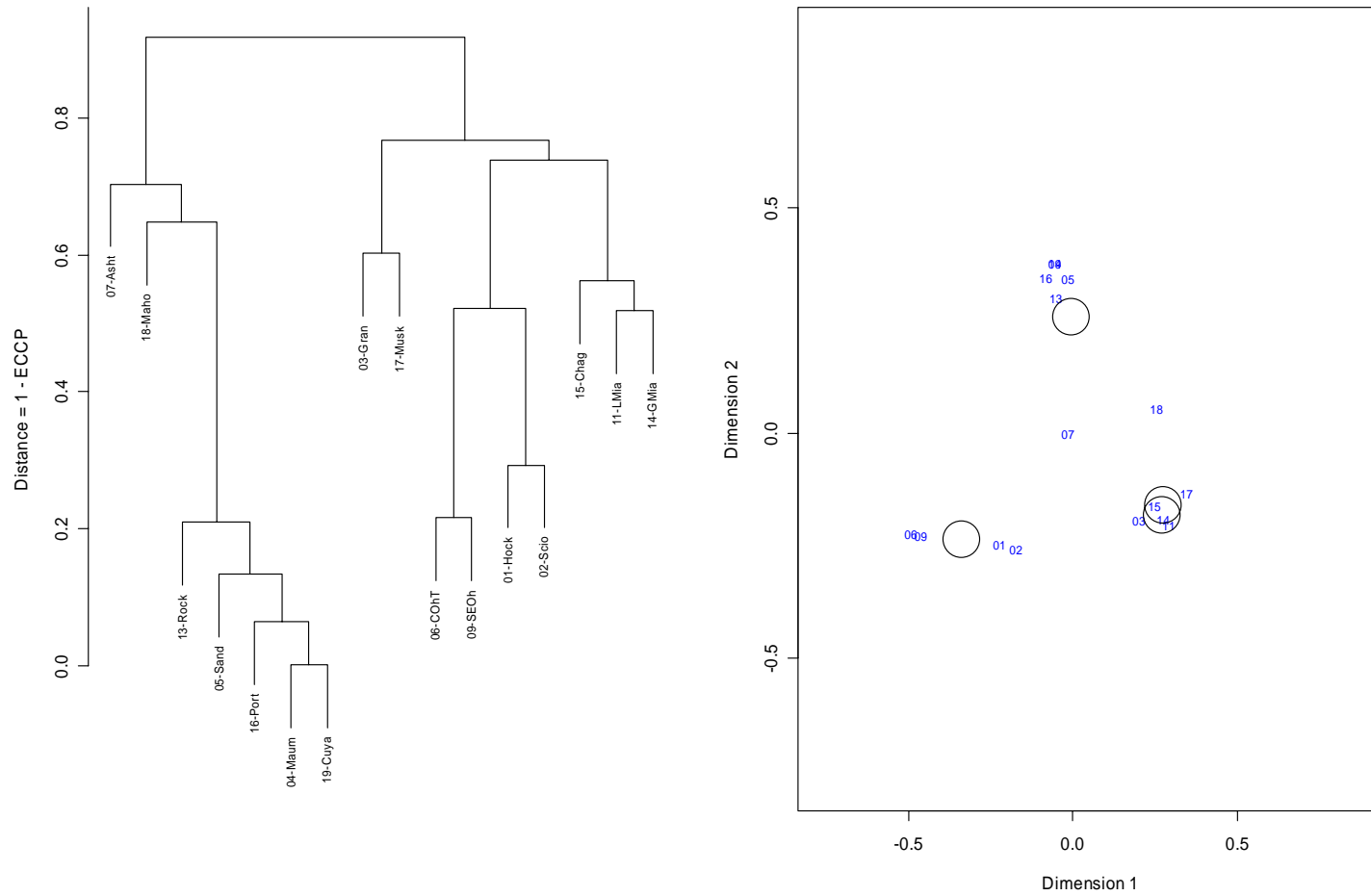


**Figure 3-26** Dendrogram and ordination plot based on ECCP from 2-class model (Ohio dataset).

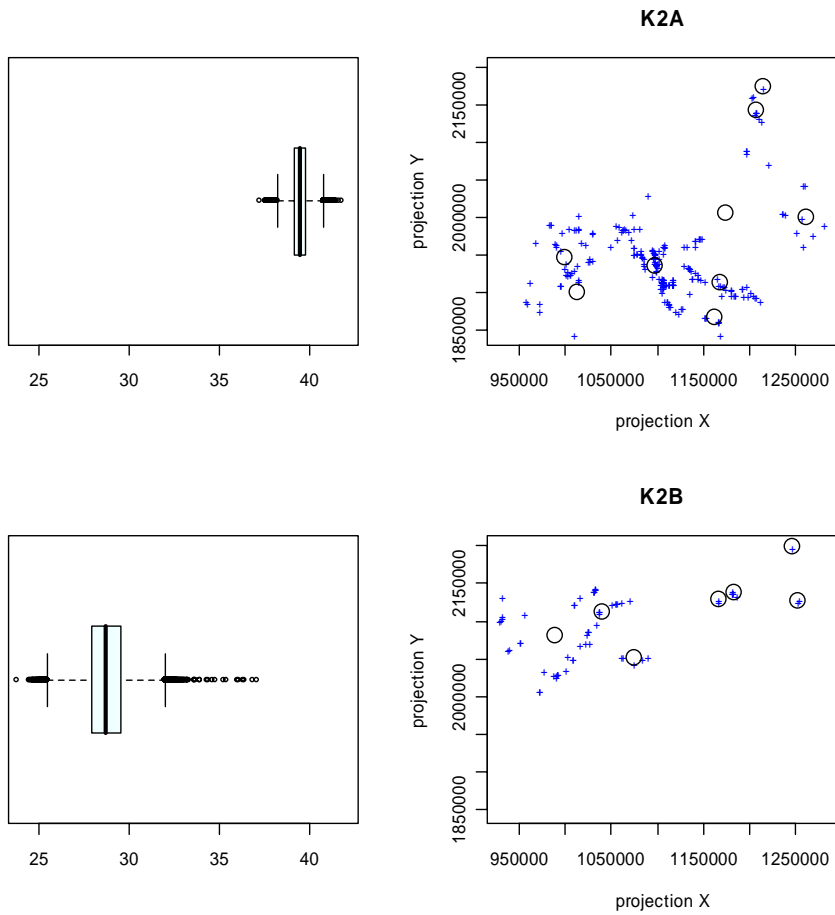




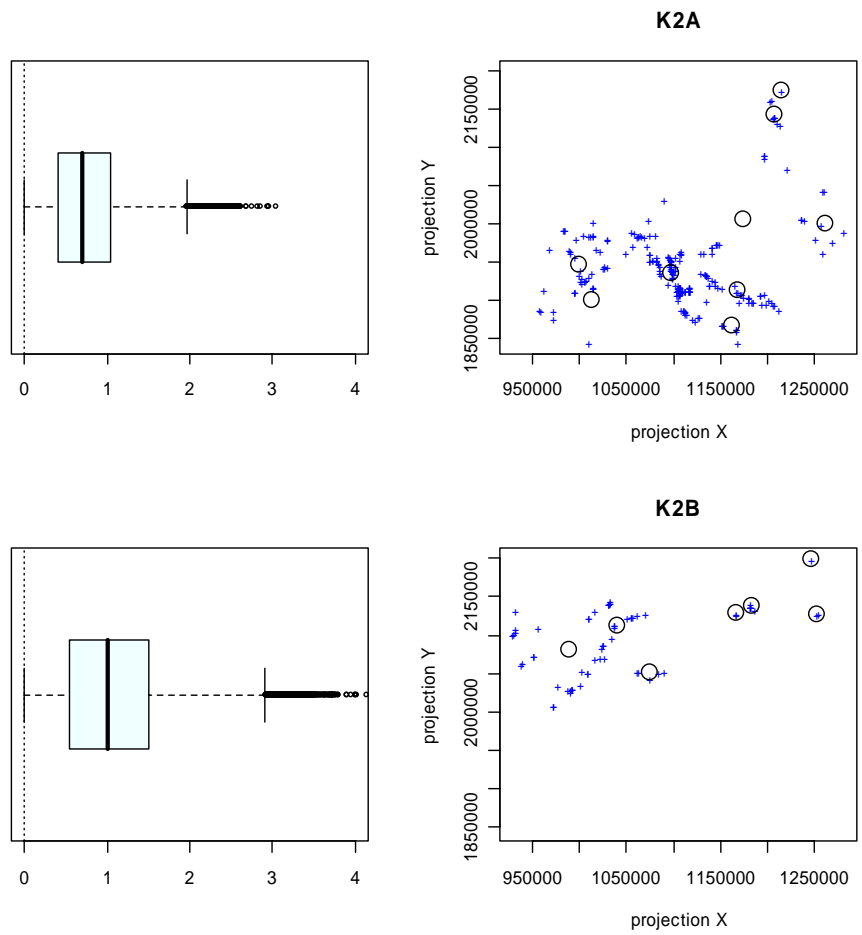
**Figure 3-27** Dendrogram and ordination plot based on ECCP from 3-class model (Ohio dataset).



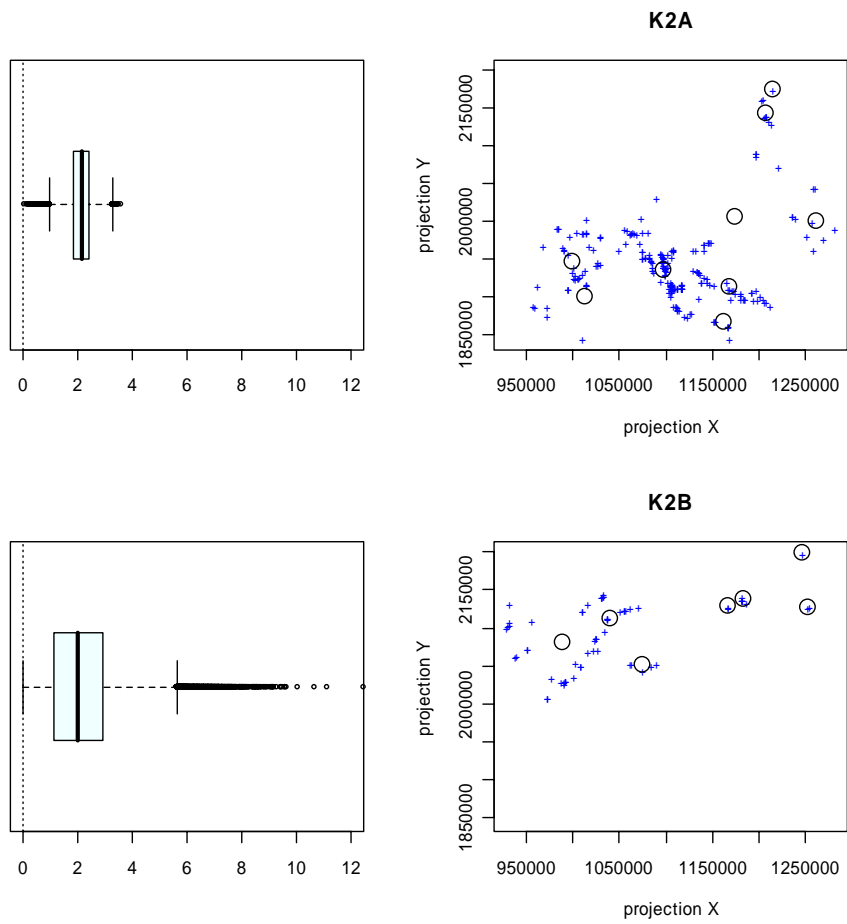
**Figure 3-28** Dendrogram and ordination plot based on ECCP from 4-class model (Ohio dataset).



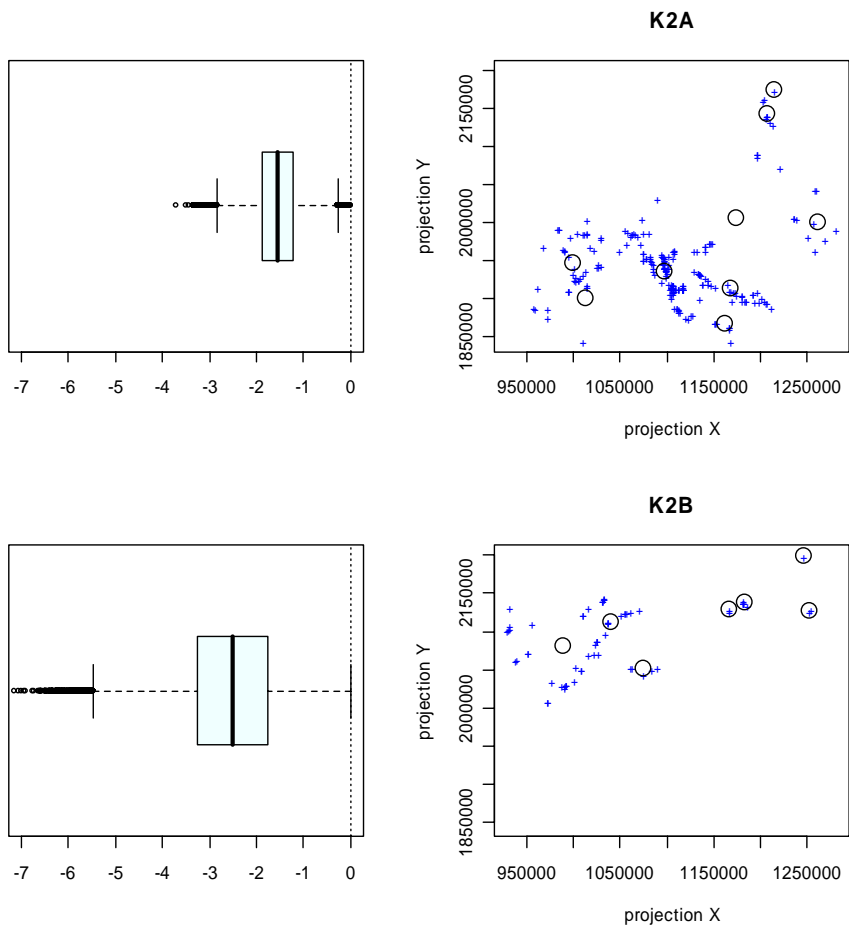
**Figure 3-29** *Estimated posterior distributions of class intercepts from 2-class model.* For each cluster, the estimated posterior distribution is depicted on the left using a boxplot. Stations assigned to a cluster mapped to the right. For the map, averaged coordinates of basins are circled.



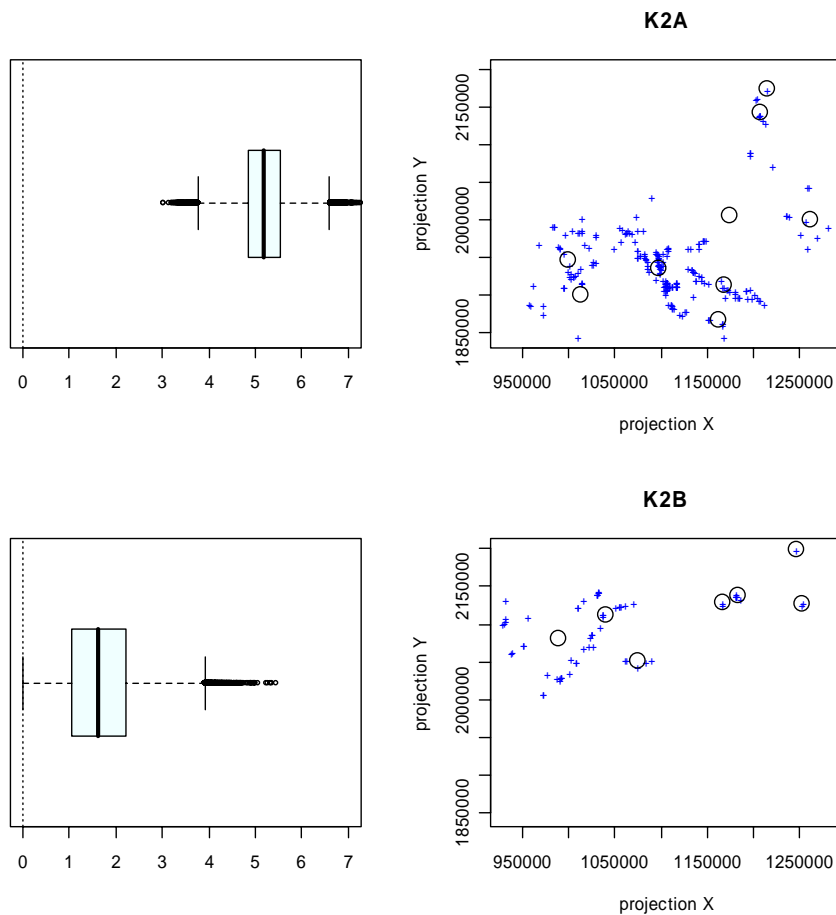
**Figure 3-30** *Estimated posterior distributions of regression coefficient for dissolved oxygen, for 2-class model. (Details are as for Figure 3-29.)*



**Figure 3-31** *Estimated posterior distributions for regression coefficient for pH, for 2-class model. (Details are as for Figure 3-29.)*



**Figure 3-32** *Estimated posterior distributions for regression coefficient for zinc, for 2-class model. (Details are as for Figure 3-29.)*



**Figure 3-33** *Estimated posterior distributions for regression coefficient for QHEI, for 2-class model. (Details are as for Figure 3-29.)*

### 3.4 DISCUSSION OF THE MODEL-BASED CLUSTERING APPROACH, WITH AN ANALYSIS ALLOWING CLASS-SPECIFIC VARIANCES

Many alternative procedures can be imagined, differing in the approach to inference, the choice of priors, approach to interactions, and so on.

We have relied initially on additive class regression models. For an economical incorporation of interactions, we suggest modeling the regression surface using a multidimensional process prior, or using radial basis functions (Fahrmeir and Tutz, 2001;

Ruppert *et al.*, 2003). However, we suggest that a relatively simple *ad hoc* approach can be based on a multidimensional LOWESS smoother, applied separately to each cluster, at each step in a Metropolis algorithm.

An important alternative is a “finite mixture” model. However, in the context of monitoring data, such a model seems to contain additional features – the class probabilities -- which do not seem clearly justified, and seem to increase the burden in explaining the model, relative to our direct assignment of probabilities to alternative classifications. (In the chapter following we observe that according to our prior, class probabilities are implicitly  $1 / k$ .)

For convenience, we have used flat priors for parameters of class regression models. The finite mixture literature is a source of diverse alternative types of priors (Wasserman, 2000). Alternatives include conjugate priors, which might shrink estimated regression coefficients towards zero, or towards a global value for the data set.

There is apparently some tendency for our MCMC chains to become trapped in parts of the model space that are associated with relatively low likelihood, a phenomenon also noted in “finite mixture” modeling (*e.g.*, Ter Braak *et al.*, 2003; Viele and Tong, 2002). Our use of selected chains will reduce uncertainty in the posterior distribution artificially, unless the excluded chains are generally associated with such low likelihood values that they can be ignored. Alternatives to the use of selected chains may involve MCMC methods with more rapid mixing, or use of some principled approach for pooling of chains (Gelman *et al.*, 1995; Viele and Tong, 2002). Selection of chains by inspection of log-likelihood graphs is the only feature not automated, in our approach.

We have provided suggestive results related to selection of the number of classes, based on the ECCP. We recommend further evaluation of the approach.

We have initially assumed a residual variance parameter, denoted  $\sigma^2$ , equal in value for each class. An alternative is to incorporation of a variance function, perhaps simply with



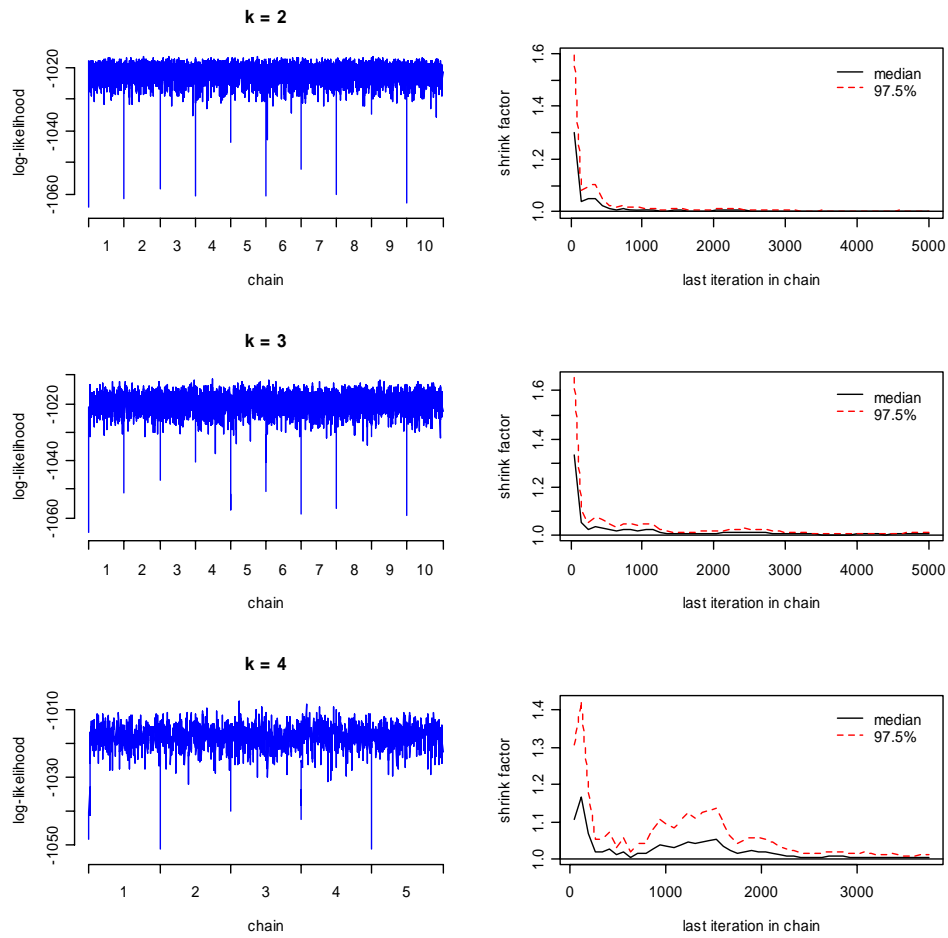
variance assumed proportional to expected response, and the constant of proportionality fitted to the data along with other parameters. In particular, in situations where a response variable is bounded, the variance is likely to be lower when the expected response is closer to a bound. Some evaluation of residuals from preliminary models has tended to confirm such a pattern.

Another possibility is to allow different variances in different classes. For the Maryland dataset, we have compared our results from a 3-class model to results from a model that is similar, except for class-specific variances  $\sigma_1^2, \dots, \sigma_k^2$ . Assuming prior independence, the posterior sampling procedure is revised so that  $\beta_j$  is updated by drawing from a multivariate normal distribution with mean vector  $(\mathbf{X}_j^T \mathbf{X}_j)^{-1} \mathbf{X}_j^T \mathbf{y}_j$  and covariance matrix  $\sigma_j'^2 (\mathbf{X}_j^T \mathbf{X}_j)^{-1}$ . The class variance  $\sigma_j^2$  is updated using

$$\sigma_j'^2 = \frac{1}{\chi_{m_j}^2} \sum_{i \in C_j'} (\mathbf{y}_i - \mathbf{X}_i' \beta_{\gamma_i}')^T (\mathbf{y}_i - \mathbf{X}_i' \beta_{\gamma_i}')$$

where  $m_1, \dots, m_k$  are counts of stations for clusters  $C_1, \dots, C_k$  according to the classification at the current iteration.

Figure 3-34 shows the log-likelihood and diagnostic plot for 10 chains generated according to this model. We computed the classification using chains 1-3, 5, and 9, chosen by inspection of the likelihood plot. The resulting classification was found to be identical to that obtained with the equal-variances assumption (Figure 3-15). Class regression results would also be identical, given the approach of basing the regression surfaces on the fixed classification, used for the Maryland analysis.



**Figure 3-34** *Ten Metropolis chains for 3-class model with heterogeneous variances (Maryland dataset). The plots give log-likelihood, convergence diagnosis with all chains, and convergence diagnosis for chains 1-3, 5, and 9.*

### 3A.1 APPENDIX: UNIT CODES AND NUMBERS OF STATIONS FOR TWO DATASETS

**Table 3A-1** Unit codes and counts of stations for two datasets.

	Unit Code <sup>1</sup>	Number of Stations	Stream or Basin Name
<b>Ohio Dataset</b>			
1	01-01-Hock	37	Hocking River
2	02-02-Scio	116	Scioto River
3	03-03-Gran	1	Grand River
4	04-04-Maum	35	Maumee River
5	05-05-Sand	5	Sandusky River
6	06-06-COht	7	Central Ohio Tributaries
7	07-07-Asht	1	Ashtabula River
8	08-09-SEOh	8	SE Ohio Tributaries
9	09-11-LMia	4	Little Miami River
10	10-13-Rock	2	Rocky River
11	11-14-GMia	40	Greater Miami River
12	12-15-Chag	8	Chagrin River
13	13-16-Port	14	Portage River
14	14-17-Musk	16	Muskingum River
15	15-18-Maho	2	Mahoning River
16	16-19-Cuya	7	Cuyahoga River
<b>Maryland Dataset</b>			
1	N-BU	12	Bush River
2	N-CK	44	Choptank River
3	N-CR	42	Chester River
4	N-EL	4	Elk River
5	N-NW	1	Nanticoke and Wicomico Rivers
6	N-PP	28	Patapsco River
7	N-PW	19	Potomac Washington Metro
8	N-PX	30	Patuxent River
9	N-WC	24	West Chesapeake
10	P-BU	12	Bush River
11	P-EL	44	Elk River
12	P-GU	42	Gunpowder River
13	P-MP	4	Middle Potomac Basin
14	P-PP	1	Patapsco River
15	P-PW	28	Potomac / Washington Metro
16	P-PX	19	Patuxent River
17	P-SQ	30	Lower Susquehanna River

<sup>1</sup>For the Ohio dataset, the 1st numeric code is assigned for our analysis; the 2nd follows Dyer *et al.* (1988).

<sup>2</sup>The first letter encodes physiographic region (N for North Coastal, P for Piedmont). The last two letters give the basin code used in the Maryland biological streams survey.

## Chapter 4

### Some Comparisons among Priors with Dirchlet-Distributed Class Probabilities

The objective of this chapter will be to gain some insights into relationships among a limited set of priors for a classification, related to the prior used in the previous chapter.

We will let  $\gamma = (\gamma_1, \dots, \gamma_n)$  denote a classification of  $n$  units into  $k$  groups, where  $\gamma_i \in \{1, \dots, k\}$  gives the class index for unit  $i$ . Some other notation related to classifications is summarized in Table 1.1.

To this point we have relied on a prior that assigned equal probability to each allowed classification. Such a prior may be denoted  $\pi_\gamma^U$ , where  $U$  stands for uniform. This definition contains no explicit references to class membership probabilities. Here we introduce class membership probabilities, which we ultimately use to embed  $\pi_\gamma^U$  in larger family of priors with Dirichlet distributions for class probabilities.

We can expect that models with Dirichlet priors for class probabilities will be applied to monitoring data. Indeed, an early application of such models was to acidity in lakes (Richardson and Green, 1997). For such models, there is interest in the most appropriate values for Dirichlet parameters. We present a limited set of empirical comparisons among priors in the family, using two datasets analyzed in the previous chapter, and a preliminary simulation study comparing sampling properties. We discuss implications of different priors, for relative cluster sizes.

The priors of the previous chapter were provisional, designed to facilitate development of applications. We suggest that in the context of analysis of monitoring data, class probabilities might represent a model feature relatively difficult to justify, compared to our assumption that partitions of the data are equally probable *a priori*. While sensitivity analyses are always desirable, the results in this chapter do not clearly point to sensitivity

to choice of prior as the most important focus of research, within the range of priors considered.

It can be argued that a better approach than that of this chapter would be to find priors based on principles such as domination by likelihood, invariance under transformations, or frequentist performance. Wasserman (2000) mentions a number of approaches from the closely related area of finite mixture modeling. However, we think there is some value in evaluating any priors likely to be used, and understanding how they are interrelated.

For the time we assume prior independence of class membership among units. This is actually violated when we require a minimum cluster size. In practice, it is convenient to implement MCMC sampling for a model without the minimum cluster size, and exclude iterations with the minimum cluster size not met. The effects of such a constraint can be quantified using simulation.

The chapter is organized as follows. In Section 4.1 we present formally a family of priors for a classification, with Dirichlet distributed class probabilities, suggested by the “finite mixture” model literature. The prior for a classification that we have used in Chapter 3, with equal probability for each classification, is a limiting case. In Section 4.2 the posterior sampling distribution used in Chapter 3 is extended to the general family. A preliminary study of sampling properties, relying on Monte Carlo simulation, is presented in Section 4.3. An empirical comparison with two datasets is presented in Section 4.4. In Section 4.5 we present some implications of the models for the relative sizes of clusters. Some technical remarks are presented in an appendix.

#### 4.1 PRODUCT-MULTINOMIAL PRIORS AND PRIORS WITH DIRICHLET DISTRIBUTED CLASS PROBABILITIES

Our definition of  $\pi_{\gamma}^U$  contained no explicit references to class probabilities. We now introduce a family of *product multinomial* priors, which incorporate class probabilities  $\tau_1, \dots, \tau_k$ , where  $\tau_j$  is the probability that the  $i$ th unit belongs to the  $j$ th class

( $i = 1, \dots, n \geq k, j = 1, \dots, k > 1$ ). Equivalently, we can say that the vector of binary indicators  $(z_{i1}, \dots, z_{ik})$ , where  $z_{ij}$  equals 1 or 0 according to whether the  $i$ th unit does or does not belong to the  $j$ th class, has a multinomial distribution with parameters 1 and  $(\tau_1, \dots, \tau_k)$ ,

$$(z_{i1}, \dots, z_{ik}) \sim \text{mult} (1, (\tau_1, \dots, \tau_k)) \text{ independently } i \text{ and } i' \neq i,$$

The prior density is  $\prod_{j=1}^k \tau_j^{n_j(\gamma)}$  where  $n_j(\gamma)$  ( $j = 1, \dots, k$ ) is the count of units in  $C_j$ , according to classification  $\gamma$ . We next embed our product-multinomial prior, in turn, in a yet larger family denoted  $\pi_\gamma^{[\alpha_1, \dots, \alpha_k]}$ , defined by placing a Dirichlet prior on the class probabilities:

$$(\tau_1, \dots, \tau_k) \sim \text{Diri}(\alpha_1, \dots, \alpha_k) \quad (4.1)$$

where  $\text{Diri}(\alpha_1, \dots, \alpha_k)$  denotes a Dirichlet distribution. The Dirichlet distribution is convenient particularly because of conjugacy to the multinomial distribution. A chapter appendix summarizes some properties of the distribution, including the probability density function and moments.

We consider only *symmetric* cases, denoted  $\pi_\gamma^{[\alpha]}$ , obtained from  $\pi_\gamma^{[\alpha_1, \dots, \alpha_k]}$  by the constraint  $\alpha_1 = \dots = \alpha_k = \alpha$ . A popular choice is  $\alpha = 1$ . Then  $(\tau_1, \dots, \tau_k)$  has a uniform distribution -- on a simplex in  $k - 1$  dimensions -- so this choice can be said to rely on a *flat* Dirichlet distribution. In particular, with 2 classes each of  $\tau_1$  and  $\tau_2$  is distribution uniformly on  $[0, 1]$ . Another possibility is  $\alpha = 1 / 2$ , associated with Jeffreys priors for finite mixture models (Wasserman, 2000). Under  $\pi_\gamma^{[\alpha]}$  the class probabilities are identically distributed with  $E(\tau_j) = 1 / k$  and  $\text{var}(\tau_j) = 1 / k (k - 1) / [k^2 (1 + k\alpha)]$ . Thus increasing the value of  $\alpha$  results in class probabilities that are close to the common mean  $1 / k$  with higher probability.

We observe that  $\pi_\gamma^U$  specifies class probabilities *implicitly*. Indeed, from the chapter appendix we have that alternative characterization, unique among the priors considered, is that  $\tau_1 = \dots = \tau_k = 1 / k$ .

A model with a hierarchy like (4.1) may sometimes be described as a “finite mixture.” However, the class probabilities  $\tau_1, \dots, \tau_k$ , which we interpret as parameters of a prior, are sometimes interpreted as frequencies, or the interpretation may be ambiguous.

## 4.2 PARTITION-SPACE RANDOM WALK ALGORITHM FOR MODEL WITH DIRICHLET CLASS PROBABILITIES

The partition-space random walk algorithm of the previous chapter is easily modified to accommodate Dirichlet distributed class probabilities. We sketch the steps of a single iteration before giving details for each step. The algorithm has been implemented for the simulation study reported in this chapter.

- (1) Use an explicit Metropolis proposal and acceptance to update the class assignment of a single unit, selected at random. Acceptance probabilities will depend on current values of  $\tau_1, \dots, \tau_k$  as well as on current values of class distribution parameters.
- (2) Sample the probabilities  $\tau_1, \dots, \tau_k$  from the Dirichlet distribution  $\text{Diri}(\alpha_1 + n_1, \dots, \alpha_k + n_k)$  where  $n_j = n_j(\gamma)$  is the size of  $C_j$  according to the current classification.
- (3) Sample values for any class-specific parameters from full conditional distributions, obtained by treating the current values for other unknowns (the classification, class probabilities) as known true values.

For the following details, we will specialize to a model with multiple-station GCU and class linear models, as in Chapter 3. Therefore our unknowns will include  $\gamma, \beta_1, \dots, \beta_k$ , and  $\sigma^2$ , as previously, and now  $\tau_1, \dots, \tau_k$  as well.

To update  $\gamma$  (Step 1) we may use the approach of Chapter 3 with a modification of the acceptance probability. Accordingly, suppose the proposal is to assign  $\text{GCU}_1$  to a new class. With unknowns other than  $\gamma$  fixed at current values the indicator vector  $(z_{11}, \dots, z_{1k})$  has a multinomial distribution with parameters 1 and  $(z_{11}^*, \dots, z_{1k}^*)$  where

$$z_{1j}^* = \frac{\tau_j f_N(\mathbf{y}_1; \mathbf{X}_1 \boldsymbol{\beta}'_j, \sigma'^2 \mathbf{I})}{\sum_{l=1}^k \tau_l f_N(\mathbf{y}_1; \mathbf{X}_1 \boldsymbol{\beta}'_l, \sigma'^2 \mathbf{I})}. \quad (4.2)$$

(As previously, primes on symbols indicate current values in an iterative scheme.) A proposal to move the unit from the  $l$ th class to the  $j$ th class will be accepted with probability

$$\min \left\{ 1, \frac{\tau_j f_N(\mathbf{y}_1; \mathbf{X}_1 \boldsymbol{\beta}'_j, \sigma'^2 \mathbf{I})}{\tau_l f_N(\mathbf{y}_1; \mathbf{X}_1 \boldsymbol{\beta}'_l, \sigma'^2 \mathbf{I})} \right\}. \quad (4.3)$$

Class regression parameters (Step 3) are updated according to the approach of the previous chapter.

We observe that if the Dirichlet parameter  $\alpha$  is taken to infinity, the class probabilities converge in probability to their common mean  $1/k$ , and (4.3) converges in probability to the acceptance probability that we used in Chapter 3. This does not seem surprising in view of relationships among these priors described in the chapter appendix.

### 4.3 A PRELIMINARY MONTE CARLO STUDY OF SAMPLING PROPERTIES

We present a small Monte Carlo experiment evaluating sampling properties associated with use of the model with Dirichlet distributed class probabilities. Comparisons are made among our priors  $\pi_\gamma^U$  and  $\pi_\gamma^{[\alpha]}$ , with different values of  $\alpha$ . Our experiment employs Monte Carlo sampling at two levels: In an outer loop we generate random data sets. For each simulated data set, we generate and summarize a posterior sample.



For the time, our model is simply a Gaussian location-difference model with two classes. For each class we assume a normal distribution for the observable variable. The mean differs in value for the two classes, while the within-class variance is equal.

We simulate data by generating a fixed number of observations for each class, varying the relative sizes of the two sets. (An alternative that may seem more appropriate when class probabilities are interpreted as frequencies would be to generate data in two stages, first generating a random classification, then sampling observations from class-specific distributions.)

For purposes of an initial Monte Carlo experiment, we evaluate inferences that do not require relabeling (see Chapter 2). We quantify two general aspects of performance, error in inferring a partition, and error in inferring the absolute difference of class means.

The simulation can be considered preliminary. Extensions could focus on a wider array of scenarios in terms of the number of classes, units per class, class-specific model, and inferences of interest.

#### *4.3.1 Procedure*

Details of the simulation are as follows.

- We assumed 16 GCU and 35 stations per GCU. This scenario resembles the data analyzed in Chapter 3, from the Maryland Biological Stream Survey (MBSS), which represent 586 stations belonging to 17 GCU. (However, the representation of GCU in the MBSS data is very unbalanced.)
- We assumed that for each station an observation is drawn from a normal distribution. The means for the two classes are assumed to differ by 1 unit. A range of values is evaluated for the within-class standard deviation  $\sigma$ , assumed equal for the two classes.

- We simulate “balanced” scenarios in which units are equally divided between two classes, and “unbalanced” scenarios with 25% of units in the smaller class and 75% in the larger.
- The minimum value of a class probability was 0.1 for models with Dirichlet distributed class probabilities.

To provide some defense against local optima, for each simulated data set we generated 3 independent chains and computed the median log-likelihood for each, using the likelihood expression of Chapters 3 and 4. Whenever the median log-likelihoods had a range of 2 or greater across chains, posterior sampling was repeated with triplets of chains, until the range of medians fell below 2. Each MCMC chain was of length 3000. The 3 chains were pooled to form the posterior sample, after discarding the first half of each. Simulations were set to run indefinitely and results were gathered at somewhat irregular intervals, resulting in different Monte Carlo sample sizes for different results, given in the right-most column of Table 4.1.

We summarize the simulation results using statistics denoted C1-C4, defined in footnotes of Table 4-1.

We evaluate inference of the partition using a statistic C1 based on the ECCP, which are arrayed in the matrix **ECCP** (Chapter 2). C1 is motivated as follows. When the partition is estimated accurately, ECCP values will be close to 1 for pairs of units that truly cluster together, and close to zero for pairs that truly cluster separately. C1 takes values in  $[0, 1]$  where a value of 1 corresponds to “perfect agreement” of the ECCP with the true partition, and a value of 0 to “perfect disagreement.”

In greater detail, let **TCC** -- for *true coclustering* -- be a square matrix that represents the true partition using binary indicators. Whereas the *i*th row and *j*th column of **ECCP** gives the fraction of the posterior sample with the *i*th and *j*th units assigned to the same class, the corresponding value in **TCC** equals 1 or 0 according to whether or not the pair truly belongs to the same class. The two matrices are compared by computing

$$\frac{\sum_{\text{cells}} [\mathbf{TCC} * \mathbf{ECCP} + (\mathbf{J} - \mathbf{TCC}) * (\mathbf{J} - \mathbf{ECCP}) - \mathbf{I}]}{n(n-1)} \quad (4.4)$$

Here the summation is over all cells in a matrix, \* indicates element-by-element multiplication of matrices, and  $\mathbf{I}$  and  $\mathbf{J}$  are an identity matrix and a matrix of 1's, each conformable for the operations required. In (4.4) the numerator contribution for a pair of units is either the estimated probability that the pair clusters separately (for a pair that actually do belong to different classes) or the estimated probability that they cluster together (for a pair that actually do belong to the same class). The denominator is the number of cells in each of our matrices, minus the number of diagonal cells. Our statistic C1 is computed by evaluating (4.4) for each simulated data set, and averaging the quantity over the Monte Carlo sample.

To evaluate inferences related to class means, we consider only inference for the label-invariant parameter  $\Delta_\mu = |\mu_1 - \mu_2|$ , which has a value of one in our experiment. This parameter is assumed estimated by the absolute difference of estimated class posterior means. We evaluate bias and mean square error of this estimator. For given data, a 95% (central) credible interval for  $\Delta_\mu$  is provided by the 2.5th and 97.5th percentiles of a posterior sample. We evaluate average width and frequentist coverage for such an interval (Criteria C3a, C3b, and C4).

Inference for  $\sigma$  was not evaluated in our experiment. However, in monitoring the simulations, estimates of the parameter are observed to track the true value well.

### 4.3.2 Results and Discussion

Results are displayed in Table 4.1 and Figure 4.1. The results do not suggest a clear preference for one or another model, because somewhat different models may be preferred basis on emphasizing different criteria of performance.

Based on results in Table 4.1, estimation of the partition, as quantified using C1, seems generally acceptable.

Figure 4.1 displays results for bias and mean square error in estimating the absolute difference of class means. The different priors are associated with similar mean square error. Bias is generally less than 15%, positive for balanced scenarios and negative for unbalanced scenarios.  $\pi_{\gamma}^U$  is associated with the least absolute bias in balanced cases and the greatest bias in unbalanced cases.

Credible intervals for the absolute difference of class means seem acceptable for use as frequentist confidence intervals, at least when the groups are well-separated. When the groups are not well separated, credible intervals are shifted such that coverage probabilities “on the upper end” (the probability of an upper bound greater than the true parameter value) exceeds the analogously-defined coverage probabilities “on the lower end.” Estimates of coverage probabilities should be based on a larger simulation.

**Table 4-1** Monte Carlo assessment of sampling error with Dirichlet distributed class probabilities.

Prior <sup>[1]</sup>	Balance <sup>[2]</sup>	$\sigma$ <sup>[3]</sup>	Performance Statistics Computed from Monte Carlo sample						
			C1 <sup>[4]</sup>	C2a <sup>[5]</sup>	C2b <sup>[6]</sup>	C3a <sup>[7]</sup>	C3b <sup>[8]</sup>	C4 <sup>[9]</sup>	$n$ <sup>[10]</sup>
Diri (0.5)	50:50	1	0.99	0.0010	0.0079	0.033	0.028	0.331	723
		2	0.76	0.0192	0.0333	0.029	0.006	0.781	617
		4	0.55	0.0533	0.1515	0.019	0.000	1.863	533
	25:75	1	0.99	-0.0047	0.0105	0.027	0.040	0.384	528
		2	0.76	-0.0392	0.0635	0.026	0.023	0.939	649
		4	0.56	-0.0094	0.1540	0.017	0.000	1.875	591
Diri (1)	50:50	1	0.99	-0.0025	0.0077	0.035	0.031	0.331	869
		2	0.77	0.0135	0.0336	0.020	0.009	0.776	922
		4	0.55	0.0926	0.1469	0.016	0.000	1.925	759
	25:75	1	0.99	-0.0027	0.0097	0.011	0.024	0.384	452
		2	0.77	-0.0354	0.0575	0.021	0.023	0.945	774
		4	0.56	-0.0382	0.1223	0.011	0.000	1.894	752
Diri ( $\infty$ )	50:50	1	0.99	-0.0029	0.0075	0.019	0.029	0.333	314
		2	0.79	0.0014	0.0366	0.033	0.023	0.744	912
		4	0.57	-0.0021	0.1282	0.017	0.001	1.666	706
	25:75	1	0.99	-0.0062	0.0097	0.023	0.033	0.387	767
		2	0.73	-0.1051	0.0708	0.014	0.088	0.893	762
		4	0.54	-0.1151	0.1278	0.011	0.003	1.614	937

<sup>[1]</sup> Symmetric Dirichlet prior for class probabilities. For other unknowns, flat priors have been assumed as described in the text.

<sup>[2]</sup> 50% of units in each class, versus 25% in one class and 75% in the other.

<sup>[3]</sup> Assumed within-class residual standard deviation.

<sup>[4]</sup> C1 = Criterion (3.2) for estimation of a partition, averaged over the Monte Carlo sample.

<sup>[5]</sup> C2a= bias in estimating  $\Delta_{\mu}$ .

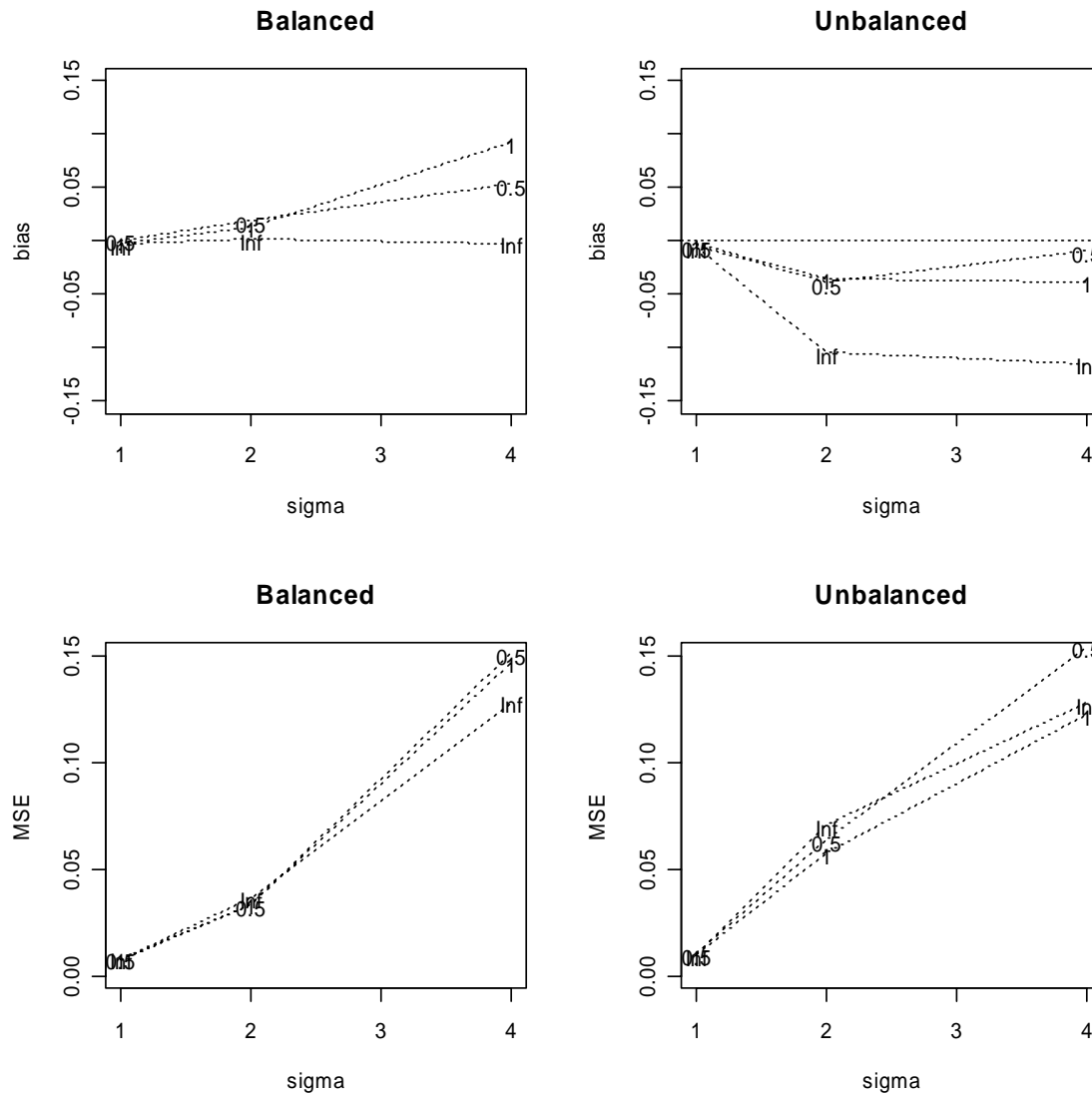
<sup>[6]</sup> C2b= mean square error in estimating  $\Delta_{\mu}$ .

<sup>[7]</sup> C3a= probability that  $\Delta_{\mu}$  falls below the 2.5th percentile of its posterior distribution.

<sup>[8]</sup> C3b = probability that  $\Delta_{\mu}$  falls above the 97.5th percentile of its posterior distribution.

<sup>[9]</sup> C4 = average width of the 95% credible interval for  $\Delta_{\mu}$ .

<sup>[10]</sup>  $n$  = Size of Monte Carlo sample (number of simulated data sets). (Simulations were set to run indefinitely and results were collected at somewhat irregular intervals.)



**Figure 4-1** Bias and mean square error results from simulation.

#### 4.4 EMPIRICAL COMPARISONS BASED ON THE MARYLAND AND OHIO DATASETS.

Here a range of priors are compared based on application to two datasets described in the previous chapter.

We consider priors  $\pi_\gamma^U$  and  $\pi_\gamma^{[\alpha]}$ , with  $\alpha = 0.5, 1, 10, \text{ and } 50$ . The minimum cluster size was 20 for the Maryland dataset and 15 for the (smaller) Ohio dataset. The expected

response was bounded to the prior range of the measured response. For the Ohio dataset, sign constraints were placed on regression coefficients. For Dirichlet priors we imposed a minimum value of 0.1 on class probabilities.

For each model we generated 10 chains with a combined length of 50,000, as in Chapter 3. Figure 4-2 displays plots of the log-likelihood for the Maryland dataset, computed using the same expression as used in Chapter 3. The plot suggests heterogeneity among chains. Our posterior sample was generated by pooling chains with median log-likelihood no smaller than -680, a value chosen by inspection of Figure 4-2. For the Ohio dataset, plots of the log-likelihood showed no indications of heterogeneity and all chains were used.

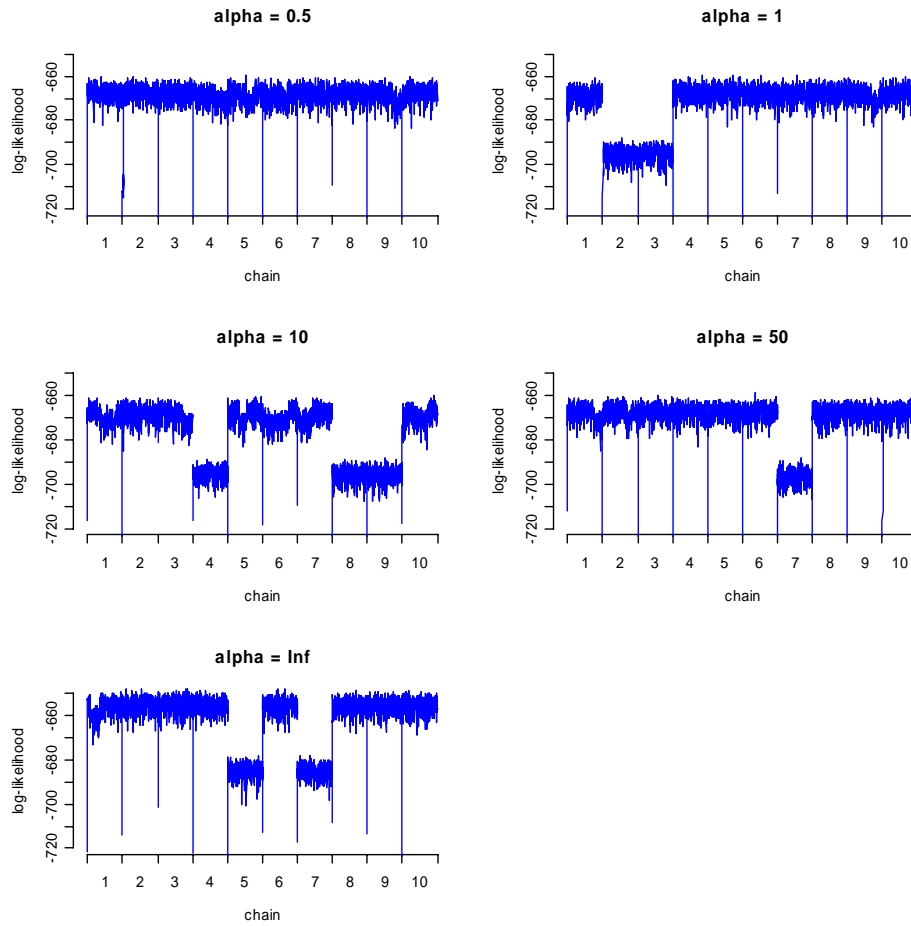
Table 4-2 displays estimated probabilities of membership in Class 1, for each unit and each model. Our estimate of the posterior probability that unit  $i$  belongs to class  $j$  is the sample fraction with unit  $i$  in class  $j$ , after relabeling the posterior sample using the algorithm of Chapter 2. For the most part the results agree well among models. Exceptions are the results for the Maryland data with  $\alpha = 10$ . In that case the log-likelihood suggests multiple solutions with similar likelihoods.

Table 4-2 suggests that assignments that are relatively confident are not apparently sensitive to the model. Indeed, apart from results with  $\alpha = 10$ , whenever a unit belongs to a given class with probability 0.8 or more according to one model the unit belongs to the same class according to other models, also with probability 0.8 or more (to a single digit accuracy).

**Table 4-2** Sensitivity of clustering results to Dirichlet parameter, based on two datasets.

	Unit Code	Number of Stations	$\alpha$				
			0.5	1	10	50	$\infty$
<b>Ohio Data</b>							
1	01-01-Hock	37	1	1	1	1	1
2	02-02-Scio	116	1	1	1	1	1
3	03-03-Gran	1	0.956	0.952	0.947	0.940	0.942
4	04-04-Maum	35	-	-	-	-	-
5	05-05-Sand	5	0.012	0.014	0.011	0.012	0.007
6	06-06-COhT	7	1	1	1	1	1
7	07-07-Asht	1	0.500	0.504	0.434	0.411	0.385
8	08-09-SEOh	8	1.00	1.00	1.00	1.00	1.00
9	09-11-LMia	4	0.956	0.951	0.941	0.939	0.938
10	10-13-Rock	2	0.129	0.116	0.090	0.076	0.082
11	11-14-GMia	40	1	1	1	1	1
12	12-15-Chag	8	0.810	0.801	0.788	0.755	0.762
13	13-16-Port	14	0.064	0.064	0.046	0.044	0.038
14	14-17-Musk	16	1	1	1	1	1
15	15-18-Maho	2	0.388	0.390	0.333	0.290	0.275
16	16-19-Cuya	7	-	-	-	-	-
<b>Maryland Data</b>							
1	N-BU	12	0.804	0.922	0.681	0.918	0.995
2	N-CK	44	1	1	1	1	1
3	N-CR	42	0.018	0.015	0.021	0.007	0.030
4	N-EL	4	0.325	0.330	0.368	0.372	0.375
5	N-NW	1	0.811	0.831	0.848	0.860	0.874
6	N-PP	28	0.078	0.086	0.060	0.085	0.084
7	N-PW	19	0.071	0.090	0.066	0.095	0.101
8	N-PX	30	0.806	0.919	0.675	0.922	0.997
9	N-WC	24	1	0.999	1	0.999	1
10	P-BU	12	0.208	0.236	0.183	0.249	0.279
11	P-EL	44	0.955	0.962	0.968	0.976	0.975
12	P-GU	42	-	-	-	-	-
13	P-MP	4	1	1	1	1	1
14	P-PP	1	-	-	-	-	-
15	P-PW	28	-	-	-	-	-
16	P-PX	19	-	-	-	-	-
17	P-SQ	30	-	-	-	-	-





**Figure 4-2** *Log-likelihood for Maryland dataset.*

#### 4.5 IMPLICATIONS FOR RELATIVE CLUSTER SIZES

The prior  $\pi_\gamma^U$  assumes equal probability for each partition. Such a prior does not assume equal probability for different cluster *sizes*. Indeed, the prior implies relatively high probability, compared to  $\pi_\gamma^{[a]}$ , for classifications that divide the units into clusters of relatively similar size. Information in  $\pi_\gamma^U$  on relative cluster sizes could conflict with information from subject matter, for example if subject-matter considerations suggests small clusters in a background of noise, or could agree with subject-matter considerations if similar-sized clusters are desirable.

Consider the random variable  $X$  representing the number of units in a Class 1, dropping the class index, and let  $\tau$  the corresponding class frequency. The expected value of  $X$  is  $n / k$ , under both  $\pi_\gamma^U$  and  $\pi_\gamma^{[\alpha]}$ . However, the variance is larger under  $\pi_\gamma^{[\alpha]}$ . Under  $\pi_\gamma^U$ ,  $X$  has a binomial distribution with variance  $n \frac{1}{k} \left(1 - \frac{1}{k}\right)$ , while under  $\pi_\gamma^{[\alpha]}$   $X$  has “extra-binomial variation”:

$$\begin{aligned} \text{var} (X) &= E_\tau \text{var} (X | \tau) + \text{var}_\tau E (X | \tau) \\ &= n \frac{1}{k} \left(1 - \frac{1}{k}\right) + n (n - 1) \text{var} (\tau) . \end{aligned} \quad (4.5)$$

The same expression holds if  $\tau$  has some other distribution with positive variance and mean  $1 / k$ . The second term can be described as an inflation of variance from treating the class frequencies as random, relative to the case with class probabilities fixed equal to  $1 / k$ .

#### 4A.1 APPENDIX: SOME PROPERTIES OF THE DIRICHLET DISTRIBUTION.

O’Hagan (1994) provides a useful overview of the Dirichlet distribution as used in Bayesian modeling (and also suggests alternative priors that can be used in the same situations). The Dirichlet density is

$$\frac{1}{B (\alpha_1 , \dots , \alpha_k)} \prod_{j=1}^k \tau_j^{\alpha_j - 1}$$

where  $B (\alpha_1 , \dots , \alpha_k)$  is a generalized beta function. The marginal distribution of  $\tau_j$  is beta with parameters  $\alpha_j$  and  $\alpha. - \alpha_j$ , where  $\alpha. = \alpha_1 + \dots + \alpha_k$ . That marginal distribution has mean

$$E (\tau_j) = \frac{\alpha_j}{\alpha.}$$

and variance

$$\text{var} (\tau_j) = \frac{\alpha_j (\alpha. - \alpha_j)}{\alpha.^2 (\alpha. + 1)} .$$

Covariances among class probabilities are

$$\text{cov}(\tau_j, \tau_l) = -\frac{\alpha_j \alpha_l}{\alpha.^2 (\alpha. + 1)}.$$

If  $(n_1, \dots, n_k)$  is an observation on a multinomial random variable with parameters 1 and  $(\tau_1, \dots, \tau_k)$ , where the latter is distributed *a priori*  $\text{Diri}(\alpha_1, \dots, \alpha_k)$ , then  $(\tau_1, \dots, \tau_k)$  is distributed *a posteriori*  $\text{Diri}(\alpha_1 + n_1, \dots, \alpha_k + n_k)$ . This suggests an interpretation of Dirichlet parameters as “prior sample sizes.” Also, the result provides a full conditional distribution that can be useful in posterior sampling.

#### 4A.2 APPENDIX: TECHNICAL REMARKS ON PRIORS FOR A CLASSIFICATION

We let  $\gamma$  denote a classification of  $n$  objects into  $k$  groups ( $k > 1, n > k$ ). We compare two priors for  $\gamma$ , namely  $\pi_\gamma^U$  and  $\pi_\gamma^{[\alpha]}$ . The former assumes that each classification is equally probable. The latter assumes that units are assigned independently, based on class probabilities  $\tau_1, \dots, \tau_k$ , which have jointly a symmetric Dirichlet distribution with parameter  $\alpha$ .

While we have initially defined  $\pi_\gamma^U$  by direct assignment of probabilities to alternative classifications, without any reference to class probabilities, an alternative description is that units are assigned independently to classes, with probability  $1/k$  of assigning a given unit to a given class. This characterization is justified by the following theorem. We will let  $H_n^k$  denote the possible classifications of  $n$  units to  $k$  groups, and  $\# H_n^k = k^n$  denote the size of that set.

**Theorem 4.1. Implicit class probabilities under  $\pi_\gamma^U$ .** Classifications in  $H_n^k$  are equally probable if and only if each unit has probability  $1/k$  of belonging to a given group, independently of other units.

**Proof.** Suppose that each unit has probability  $1 / k$  of belonging to each group independently of other units. Then the probability is equal to  $(1 / k)^n$  for each classification. Conversely, suppose each classification is equally probable. Consider the probability that Unit 1 belongs to  $C_j$ . This equals  $\# H_{n-1}^k / \# H_n^k = 1 / k$  because  $H_{n-1}^k$  represents the (equally probable) possibilities for assigning the remaining  $n - 1$  units -- those other than Unit 1. Therefore Unit 1 has equal probability of belonging to each group. Also, assignments are independent among units: Consider an event that gives the classes for some subset of units,  $x$  in number. The event has probability

$$\# H_{n-x}^k / \# H_n^k = (1 / k)^x .$$

This is a product of positive valued functions of  $x$  assignments of individual units, which are therefore independent according to a basic probability result. (QED)

It is interesting to consider the range of priors that share with  $\pi_\gamma^U$  the property that classifications are equally probable. For any classifications with the same counts per class,  $n_1, \dots, n_k$  say, the probabilities are equal to

$$\int_{\tau_1, \dots, \tau_k} \prod_{j=1}^k \tau_j^{n_j(\gamma)} \pi_\tau (\tau_1, \dots, \tau_k) d\tau_1, \dots, d\tau_k \quad (4.5)$$

under  $\pi_\gamma^{[\alpha_1, \dots, \alpha_k]}$ . However:

**Theorem 4.2.** Under  $\pi_\gamma^{[\alpha]}$  some classifications have unequal probability.

**Proof.** If a prior assigns equal probability to each classification, an implication of Theorem 4.1 is that  $n_1$  has a binomial distribution with parameters  $n$  and  $1 / k$ . However, the variance of such a random variable is smaller than the variance under  $\pi_\gamma^{[\alpha]}$ , given by Expression (4.2).

**Theorem 4.3.** As  $\alpha$  is taken to infinity through positive values,  $\pi_\gamma^{[\alpha]}$  converges in distribution to  $\pi_\gamma^U$ .

**Proof.** Write

$$\Pr(\gamma) = \int_{\tau_1, \dots, \tau_k} f(\tau_1, \dots, \tau_k) \pi_\tau(\tau_1, \dots, \tau_k) d\tau_1, \dots, d\tau_k$$

where  $f(\tau_1, \dots, \tau_k) = \prod_{j=1}^k \tau_j^{n_j(\gamma)}$ . The class probabilities  $\tau_1, \dots, \tau_k$  converge in probability to  $1/k$  given the moments of the Dirichlet distribution in the previous appendix. Given continuity of  $f$ , the function converges in probability to  $(1/k)^n$ . The theorem follows from the Lebesgue dominated convergence theorem (Billingsley, 1995; Rao, 1973). For the dominating function, we may use the function with constant value 1.

The following heuristic seems helpful. Write  $\Pr(\gamma) = E_{\tau_1, \dots, \tau_k}(f(\tau_1, \dots, \tau_k))$ . The claim is simply that as each of  $\tau_1, \dots, \tau_k$  converges in probability to the common mean  $1/k$ , the mean of  $f(\tau_1, \dots, \tau_k)$  converges to  $f(1/k, \dots, 1/k)$ .

## Chapter 5

### Clustering Monitoring Stations Based on Two Rank-Based Criteria of Similarity of Temporal Profiles

While the previous chapters have focused on grouping stations according to similarity of multiple regressions, here the focus is grouping stations based on similarities of temporal profiles, for water quality variables that have been measured at multiple points over time, with a view towards understanding regional patterns.

An important use of water quality data is to evaluate trends over time (Gilbert, 1987; Helsel and Hirsch, 1992; Millard and Neerchal, 2000). Information on temporal trends may be useful in formulating management strategies to maintain and improve water quality. In practice, statistical trend evaluation may involve separate analyses for individual monitoring stations. To interpret the volume of statistical results from a station-by-station analysis with multiple water quality endpoints may be challenging. Multivariate and graphical techniques are likely to be helpful, by revealing patterns in data representing multiple stations. Information on regional patterns may be useful in formulating regional management strategies.

Hierarchical clusters analysis will be used to group monitoring stations according to similarities in temporal patterns. In choosing a specific hierarchical cluster analysis procedure, important decisions include a criterion of similarity among the temporal profiles, and an algorithm that can use the chosen criterion to identify groups of stations. (We will sometimes use the term *similarity* for brevity, while recognizing that some calculations actually require a measure of *dissimilarity*.) We focus primarily on the problem of quantifying similarity among temporal profiles, while relying on standard hierarchical cluster analysis software to perform clustering. We use two different clustering algorithms, depending on the choice of a criterion of similarity among temporal profiles.

Our similarity criteria will be rank-based, yielding the same results when applied to ranks of measurements – where ranking is carried out separately within the series for individual stations – as when applied to the original measurements. Rank-based procedures are often used with environmental monitoring data, particularly because of outliers and non-detected constituents (Gilbert, 1987; Helsel and Hirsch, 1992; Millard and Neerchal, 2000). Thus our reliance on rank-based procedures complements current practice in environmental trend assessment.

We compare two rank-based criteria of similarity. A criterion described in Section 5.1.2 is related to a test for trend homogeneity that is standard in environmental trend assessment (van Belle and Hughes, 1984). Data for a station are reduced to a scalar summary statistic, and clustering may be based on differences in values of the statistic between pairs of stations. The effect is that stations are grouped according to the strength and sign of trend. Because of a relationship to the homogeneity test, hierarchical cluster analysis based on this criterion may complement standard trend evaluations.

However, we suggest that reduction of the data for a station to a scalar summary may result in lower sensitivity to patterns with some practical importance, such as a reversal in trend, in a particular a region. Therefore in Section 5.3 we propose an alternative criterion based on concordance of changes in the water quality endpoint between pairs of stations, from one measurement time to the next.

Compared to the methods in previous chapters, we will rely more on relatively standard approaches to cluster analysis. Summaries are provided by general texts such as Gordon (1999), Romesburg (1984), Seber (1984), and Venables and Ripley (1994). To avoid repetitious citation, we will simply note that where we do not give a specific reference for an aspect of cluster analysis, the aspect is covered in Seber (1984) and most other general texts.

## 5.1 CLUSTER ANALYSIS METHODOLOGY

### 5.1.1 *Hierarchical Clustering of Monitoring Stations Based on Similarities in Temporal Profiles*

Our objective will be to group monitoring stations based on similarities in temporal profiles, when a water quality endpoint has been measured at a series of points in time at each station. If each of  $n$  measurement stations is measured at each of  $T$  measurement times, then we may let  $x_{it}$  denote the  $t$ th measurement for the  $i$ th station. In practice, as in applications presented here, measurements may be missing for some measurement times, for some stations.

Cluster analysis is one of several multivariate techniques for evaluation of information in the form of an index of pairwise similarity or dissimilarity among  $n$  objects (Seber, 1984). If  $d_{ij}$  denotes the dissimilarity of objects  $i$  and  $j$ , reasonable requirements are  $d_{ij} \geq 0$ ,  $d_{ii} = 0$ ,  $d_{ij} = d_{ji}$  (Seber, 1984; Venables and Ripley, 1998). It is convenient to arrange dissimilarity values in an  $n \times n$  matrix, where the cell in the  $i$ th row and  $j$ th column gives  $d_{ij}$ . For our purposes it is useful to employ what may be the simplest approach, namely hierarchical cluster analysis. We have used the R library function `hclust`, which provides seven linkage criteria as of Version 2.3. Of linkage criteria that are relatively standard, we have relied on the *average linkage approach* and *Wards method*, depending on the criterion for comparing temporal profiles, as will be explained.

Clusters can be extracted from a dendrogram by specifying a number of clusters, or a critical dissimilarity. An important issue in the application of cluster analysis is specification of the number of clusters. The clusters will tend to be more homogeneous internally if more are extracted, but extracting more clusters will not necessarily result in a more useful classification. As an alternative to specifying the number of clusters one may seek to identify individual clusters that are in some sense well supported, without necessarily committing to an exhaustive classification.



The procedures that we discuss have been programmed in R (R Core Development Team, 2005). In addition to using `hclust` to generate dendrograms, we use `cutree` to extract clusters from a dendrogram object.

### *5.1.2 A Criterion of Trend Similarity Related to a Standard Trend Homogeneity Test*

Van Belle and Hughes (1984) introduced a chi-square statistic for testing homogeneity of environmental trends. Their objective of trend homogeneity testing is evidently closely related to our objective of using cluster analysis to find homogeneous groups of stations. A rank-based statistic  $Z$  is computed for each station. When the objective is to test the trend for one station, a  $p$ -value is computed by referring  $Z$  to a standard normal distribution. The homogeneity of trend test combines the  $Z$  statistics representing  $n$  stations,  $Z_1, \dots, Z_n$  say.

Before describing the trend homogeneity test in greater detail, we summarize computation of the  $Z$  statistic for a single station. The statistic is related to Kendall's  $\tau$  (Kendall, 1962) as used for relating water quality endpoint to time, involving consideration of each pair of measurement times. We suppose that the water quality endpoint is measured at each of  $T$  times. From the  $T$  measurements we compute

$$Z = (S + \delta) / \sqrt{\text{var}(S)}.$$

$S$  is computed by considering each of  $T(T - 1) / 2$  distinct pairs of measurement times. Each of these pairs is scored as “tied,” “concordant,” or “discordant,” according as the water quality endpoint is equal in value for both measurement times, increases in value from the first to the second for both, or decreases from the first to the second for both. (The terminology reflects whether or not the sign of change for the water quality endpoint agrees with the sign of change of the time variable.) Ties are possible because of limited measurement precision, or because non-detects are present in the data and counted as ties.  $S$  is computed by subtracting the number of discordant pairs from the number of concordant pairs. The value of  $\delta$  is  $-1$ ,  $0$ , or  $1$  according as  $S$  is positive, equal to  $0$ , or negative.  $\text{var}(S)$  is an estimate of sampling variance under an assumption of no trend,

in our applications incorporating a standard adjustment for tied measurements (Gilbert, 1987, Expression 17.2).

For the test of homogeneity, the test statistic is computed as  $\chi_H^2 = \sum_{i=1}^n (Z_i - \bar{Z})^2$  where  $\bar{Z}$  is the average of  $Z_1, \dots, Z_n$ . To compute a  $p$ -value, for testing a null hypothesis of no trend heterogeneity,  $\chi_H^2$  is referred to a chi-square distribution with  $n - 1$  degrees of freedom. A statistically significant result is taken as evidence for trends differing in sign or magnitude among stations. It can be helpful to observe that  $\chi_H^2$  is a sum of squared deviations from a mean (a corrected sum of squares or  $SS_c$ ), considering that some cluster analysis procedures are designed to minimize  $SS_c$ .

When our interest is in finding homogeneous groups of stations, the test may be applied to individual groups, or to the combined data. Also, a straightforward extension allows us to test a null hypothesis of simultaneous trend homogeneity of  $k$  groups of stations, against an alternative that one or more of the groups is heterogeneous (*e.g.*, Table 1). The value of the chi-square test statistic is obtained by computing  $\chi_H^2$  for each group, as for testing homogeneity of the group, and summing across groups. The  $p$ -value is computed by referring the statistic to a chi-square distribution with  $n - k$  degrees of freedom.

Here our objective is to use  $Z$  statistics to group monitoring stations.  $\chi_H^2$  provides a criterion that can be used to identify homogeneous groups. The dissimilarity of the  $i$ th and  $j$ th stations can be taken to be the squared difference of  $Z$  scores  $(Z_i - Z_j)^2$ . We will denote this criterion  $ZDiffs^2$  (for Squared  $Z$  differences). Some justification is provided in the next section. A well-known manipulation results in a useful relationship between  $\chi_H^2$  and the sum of the  $ZDiffs^2$  across pairs of stations:

$$\sum_{i=1}^{n-1} \sum_{j=i+1}^n (Z_i - Z_j)^2 = \sum_{i=1}^{n-1} \sum_{j=i+1}^n (Z_i - \bar{Z} + \bar{Z} - Z_j)^2 = n \times \chi_H^2.$$

For better or worse, the computation of  $ZDiffs^2$  can be carried out for a pair of stations with measurement intervals that do not overlap. In case of missing measurement times for some stations, it may be sensible to include, among criteria for selection of stations, some criterion of minimum overlap in measurement periods.

### 5.1.3 Clustering with Ward's Method and Transformed Plotting Heights

To perform clustering based on the  $ZDiffs^2$  criterion, we use the incremental sum of squares method (also known as Ward's method) because of a clear relationship to our objective of finding groups of stations homogeneous according to  $\chi_H^2$ . Let  $\chi_H^2(C)$  denote the test statistic for trend homogeneity, computed using stations in a set  $C$ , but with a value of zero if  $C$  contains only one station. The increment from merging two clusters  $C_i$  and  $C_j$  is

$$\Delta\chi_H^2(C_i, C_j) = \chi_H^2(C_i \cup C_j) - \chi_H^2(C_i) - \chi_H^2(C_j).$$

According to Ward's method, as expressed for our context, the pair of clusters merged at a given step is such as to minimize such an increment. It is clear that the effect at a given iteration is also to minimize the chi-square statistic, described in the previous section, for a test of simultaneous trend homogeneity.

In a default dendrogram based on Ward's method (*e.g.*, Figure 5-4), dendrogram branch-points are plotted at twice the  $\chi_H^2$  increment (Seber, 1984). The effect of the factor of 2 is that when a cluster includes only two stations, the height plotted equals  $ZDiffs^2$ . We find it useful to modify the dendrogram by substituting plotting heights more closely related to trend homogeneity testing. Cluster  $C$  is plotted at height  $\sqrt{\chi_H^2(C) / (n_C - 1)}$  where  $\chi_H^2(C)$  is the chi-square statistic and  $n_C$  the number of stations for  $C$ . We observe that for the test of homogeneity of  $C$ , the expected value of the chi-square distribution equals its degree of freedom  $n_C - 1$ . Therefore values of our modified plotting height greater than 1 are larger than expected under an assumption of homogeneity. (Of course, such a distributional assumption does not take into account

that our groups are defined so as to minimize chi-square increments.) We apply the square-root transformation in view of the skewness of the  $Z\text{Diffs}^2$  distribution over pairs of stations, which without our transformation results in very short relative heights for the first clusters joined. Figure 5-5 displays an example of the modified dendrogram, which may be compared to the default approach, displayed in Figure 5-4.

For such a modification of a dendrogram, it is convenient to work in a programming environment where a dendrogram object can be generated using a library function and then modified, and the modified dendrogram plotted or otherwise evaluated. This allows modification of selected features (such as plotting heights in our approach) while other aspects of the dendrogram specification are handled by the library routine. For some manipulations of dendrograms, it is convenient to rely on recursive functions. For example, the count of stations for a cluster is the sum of counts for its member clusters. Similarly, cluster  $\chi_H^2$  values are amenable to recursive computation, based on chi-square increments generated using Ward's method. Recursive functions, and the capability for generating and manipulating dendrogram objects, are features of R in particular.

#### *5.1.4 An Approach Based on Concordant and Discordant Changes over Time*

For a second criterion of similarity among temporal trends, which we think may be sensitive to a wider array of temporal profile similarities, we propose the Rank Temporal Profile Similarity Index (RTepsi). For a given pair of stations, the value of the index is based on concordance of temporal changes between stations.

Suppose we have  $T_{ij}$  measurement times with measurements for both of stations  $i$  and  $j$ . Let  $x_{i1}, \dots, x_{iT_{ij}}$  denote the measured values for station  $i$  and  $x_{j1}, \dots, x_{jT_{ij}}$  measured values for station  $j$ . In the sequel, for brevity, we will let  $T$  denote the number of time-points, with the understanding that in case of missing measurements at some times, this may actually depend on the pair of stations to be compared. (The symbol  $x_{it}$  used in the previous section now has no definite meaning. For example  $x_{it}$  in the comparison of station  $i$  to station  $j$  need not equal  $x_{it}$  in the comparison to station  $j'$ .)

We will say that water quality endpoint changes for the two stations are “concordant” for a pair of measurement times when the measured value increases from the first time to the second for both stations, decreases for both, or is unchanged for both. A pair of measurement times will be called “discordant” for two stations if not concordant. Ties in particular are counted as concordances. Then our measure of similarity for a pair of stations is the fraction of pairs concordant, out of the  $T(T - 1) / 2$  pairs with measurements for the two stations. Our index can be expressed formally as

$$RTepsi_{ij} = \frac{1}{T(T - 1) / 2} \sum_{k=1}^{T-1} \sum_{l=k+1}^T \left[ I \left( (x_{il} - x_{ik}) (x_{jl} - x_{jk}) > 0 \right) + I \left( x_{il} = x_{ik} \right) I \left( x_{jl} = x_{jk} \right) \right]$$

where the indicator function  $I(c)$  equals 1 or 0 according as condition  $c$  does or does not hold.

Where a criterion of dissimilarity is required -- rather than a criterion of similarity -- we subtract RTepsi values from one. The result can be described as the fraction of pairs of measurement times that are discordant, relative to the number of pairs with data available.

In our implementation, computation of RTepsi for a pair of stations is subject to a minimum count of years with measurements for both stations. To impose this minimum count we first form a matrix with RTepsi values, including all stations with some measurements available. For any pair of stations with too few times measured for both, the missing value code is entered in the appropriate cell of the matrix. Stations are then deleted one at a time until the matrix contains no missing values, at each step deleting the station associated with the largest number of values missing.

To perform hierarchical cluster analysis, we use the average linkage criterion, which is relatively conventional and easily explained.

## 5.2 AN EMPIRICAL COMPARISON OF THE TWO CRITERIA

### 5.2.1 *Measurements of Dissolved Oxygen in the James River of Virginia*

The Virginia Department of Environmental Quality (VADEQ) samples water quality from streams and rivers in Virginia (USA), for evaluation of water quality status and trends. We report comparisons of the two criteria based on measurements of dissolved oxygen (DO,  $\text{mg}\cdot\text{L}^{-1}$ ) from 71 stations on the James River and its tributaries, that met criteria for computation of both measures of similarity.

The data used are limited to the 15-year series ending in 2004. Stations were included such that, for each pair of stations included, there were 5 or more years with measurements for both stations in the pair, within the 15-year series. Figures 5-1 and 5-2 display coordinates for 71 monitoring stations included in analyses.

Most stations had multiple measurements in some years. Before computing similarity criteria, the data were reduced to annual median values, resulting in at most one concentration for a given combination of station and year.

Information available to us includes a grouping of stations according to “streams,” distinguished using a system of 3-letter codes. Of the 71 stations included, 20 were actually located on the James River (JMS). Other “streams” correspond to tributaries. The tributaries are represented very unequally, with 11 stations for the Pagan River (PGN), 2-6 for 5 other tributaries, and a single station each for 22 tributaries.

Appendix 1 lists the 71 stations contributing data to our analysis, according to station codes used by VADEQ. For labeling some graphs, we have relied on a system of more compact station codes incorporating only the stream code and a numeric index of stations within streams. Stations are numbered according to distance from the mouth of a stream, based on distances embedded in the VADEQ codes. The appendix displays the correspondence between the two systems, allowing retrieval of detailed information for any station of interest, from the VADEQ (2006) web site.

Our initial analysis involved clustering the set of 71 stations. The analysis using all stations was followed by more focused analysis using Pagan River stations, after detecting an interesting regional pattern involving those stations.

### *5.2.2 Results*

An exploration of regional patterns in temporal profile information may naturally include separate graphical analyses for some pre-defined subsets of the data, such as river basins or physiographic provinces. Figure 5-3 displays temporal profiles for annual medians of Pagan River and JMS, which together account for almost half of the stations used in our analyses. These results may be compared to some subsequent graphs displaying profiles for clusters found using our hierarchical cluster analysis procedures. In particular an apparent negative trend for stations associated with the Pagan River appears repeatedly in subsequent analyses.

Dendrograms based on the two similarity criteria are displayed in Figures 5-4 to 5-6. A consistent feature is a cluster with a relatively large number of stations from the Pagan River. Also, the single station for James Creek (JOG-1 or 2-JOG000.62), located at the confluence with the Pagan River, clusters relatively closely with Pagan River stations. The Pagan River station with apparently the most distinct profile (PGN-11 or 2-PGN010.07) happens to be the Pagan River station furthest upstream.

Apart from the presence of a largely PGN cluster, the dendrograms do not display conspicuous similarities. The two criteria are not strongly correlated across pairs of stations (Figure 5-7).

Figure 5-8 displays hierarchical cluster analysis results obtained using only the Pagan River and Jones Creek stations. (The coordinates of these stations, which are relatively closely spaced, are displayed in Figure 5-2.) Figure 5-9 compares the profiles for two stations of particular interest (JOG-1, PGN-11) to the combined set of Pagan River and Jones Creek profiles. The profile for PGN-11 is seen to be relatively flat, compared to

other Pagan River profiles. An apparent difference between dendrograms generated according to the two criteria is that using the RTepsi criterion there is a greater tendency for adjacent stations to cluster together. (Again, note that stations are numbered from the mouth towards headwaters.)

In selecting the number of clusters supported by the data, one may consider the results from the pooled homogeneity test, with pooling over the clusters. In Table 1 we display some results from the pooled test, with different numbers of clusters. The conventional test of van Belle and Hughes, with 70 degrees of freedom, corresponds to the case of a single cluster ( $k = 1$ ). The statistically significant result ( $p \leq 0.01$ ) provides support for an effort to identify patterns. Statistical significance disappears with two or more clusters based on the  $ZDiff^2$  criterion, while with the RTepsi criterion the test is significant with up to 3 clusters (suggesting at least 4 clusters). We expect that, with  $k \geq 2$ , true false positive rate of the test will be lower than the nominal rate particularly with the  $ZDiffs^2$  criterion, which identifies clusters so as to minimize the test statistic.

Figures 5-10 and 5-11 display temporal profiles for clusters extracted from each dendrogram, somewhat arbitrarily assuming 4 clusters. Each figure provides some support for a cluster with a relatively large number of stations on the Pagan River, represented in panel 4 of Figure 5-10 and panel 3 of Figure 5-11. Coordinates of stations assigned to the 4 clusters are mapped in Figures 5-12 and 5-13. The clusters display little indication of regional patterns.



**Table 5-1** *Homogeneity of trend tests with pooling over clusters.*  
(Clusters have been identified based on two different hierarchical cluster analysis procedures.)

<i>k</i>	degrees of freedom	ZDiff <sup>2</sup> Criterion		RTepsi Criterion	
		pooled chi-square statistic	<i>p</i> -value	pooled chi-square statistic	<i>p</i> -value
1	70	182.2	<0.01	182.2	<0.01
2	69	45.1	0.99	130.6	<0.01
3	68	21.9	1.00	95.3	0.02
4	67	13.8	1.00	73.5	0.27
5	66	6.0	1.00	64.0	0.55

### 5.3 DISCUSSION

The most pronounced pattern that we have detected, reflecting data for multiple stations, is a negative trend of DO for stations associated with the Pagan River and Jones Creek (PGN/JOG). The density of sampling stations for these streams is relatively high because of specific water quality concerns. Therefore it can be argued that a specific cluster for these streams is an artifact of high sampling density. Another region might also be associated with a cluster, if sampled with similar density. Nevertheless, we think the results support that the methods provide efficient recognition of the most important patterns.

Given that the Pagan River and Jones Creek empty into the James River estuary and nearby points, with neither emptying into the other, the correlation between the two streams may reflect the action of tides.

Low DO can be associated with ecological degradation. However, the results for PGN/JOG do not establish an ongoing pattern of degradation. The pattern observed might reflect efforts to control nutrient enrichment, which often has the role of an ecological stressor. For practical reasons, measurements are taken during daytime. In presence of high solar irradiation, nutrient enrichment may enhance photosynthetic activity and lead to high DO. Such an increase in DO may be transient particularly in a

warm stream. The negative trends for PGN/JOG may be partly due to early spikes in DO, and the profiles may now be stable (Jason Hill, personal communication). Information on diurnal variation may be valuable.

Our limited empirical comparison does not seem to support a strong preference between the two criteria of temporal pattern similarity. However, in practice the relatively limited specificity of the  $Z\text{Diffs}^2$  criterion could be important in some situations. Figure 5-14 displays a hypothetical example where three stations have identical  $Z$  statistics, although the temporal profiles differ in ways that could be important, if such a pattern is encountered in practice. In practice, a change in the sign of trend, as for Station A, might reflect a change in land use or introduction or removal of a source of pollution. In general, we expect that  $Z$  statistics are not reliable for capturing regional patterns in non-monotone profiles. Non-monotone, regional profiles potentially include local effects of climatic fluctuations, as well as changes in the sign and magnitude of trends. Results from separate analysis with Pagan River and Jones Creek data are consistent with a conclusion that the  $\text{RTepsi}$  criterion may capture more information in the temporal profiles, relevant for grouping stations.

As is often the case in applications of hierarchical cluster analysis, inspection of the dendrograms displayed in Figures 5-5 and 5-6 does not lead to optimism in the possibility of finding a simple, automated procedure for determining the number of clusters. While a cluster with a large number of Pagan River stations is well supported, an automated procedure that would identify such a cluster might identify additional clusters that are not as well supported.

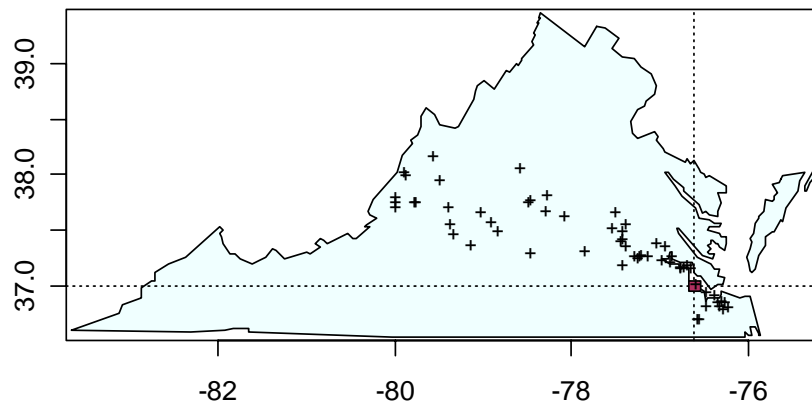
Figure 5-15 suggests a plausible approach for selecting the number of clusters when relying on hierarchical cluster analysis with the  $\text{RTepsi}$  criterion. The average within-cluster value, viewed as a measure of within-cluster homogeneity, is plotted against the number of clusters for 1-10 clusters. Our average is computed in two stages, first averaging over pairs of stations within each cluster, then averaging the results from the first step, over clusters. Formally, for  $k$  clusters  $C_1, \dots, C_k$ , we compute

$$\frac{1}{k} \sum_{l=1}^k \left( \frac{1}{N_{C_l}} \sum_{(i,j) \in C_l} \text{RTepsi}_{ij} \right)$$

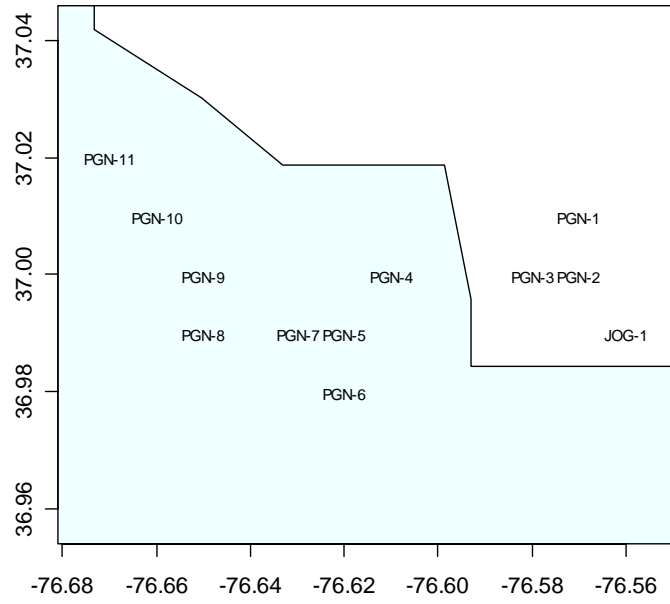
where  $N_{C_l}$  is the number of stations in  $C_l$ . It is no surprise that homogeneity as measured increases with the number of clusters. However, if the curve had showed evidence of a plateau the graph might have been taken to suggest a number of clusters, considering the use of analogous plots in multivariate analysis and statistical modeling. Unfortunately, the approach does not seem to suggest a definite number of clusters in our case. A possible improvement might incorporate a penalty for increasing the number of clusters, perhaps based on an expected increase in averaged similarity.

We have focused on patterns in the data for a single water quality endpoint measured at multiple points in time. Trend evaluation is typically required for multiple measured variables as well as for multiple stations. Therefore we think it is desirable to explore multivariate techniques designed to simultaneously evaluate multiple measurements, particularly rank-based procedures (Lettenmeier, 1976; Rheem, 1992).

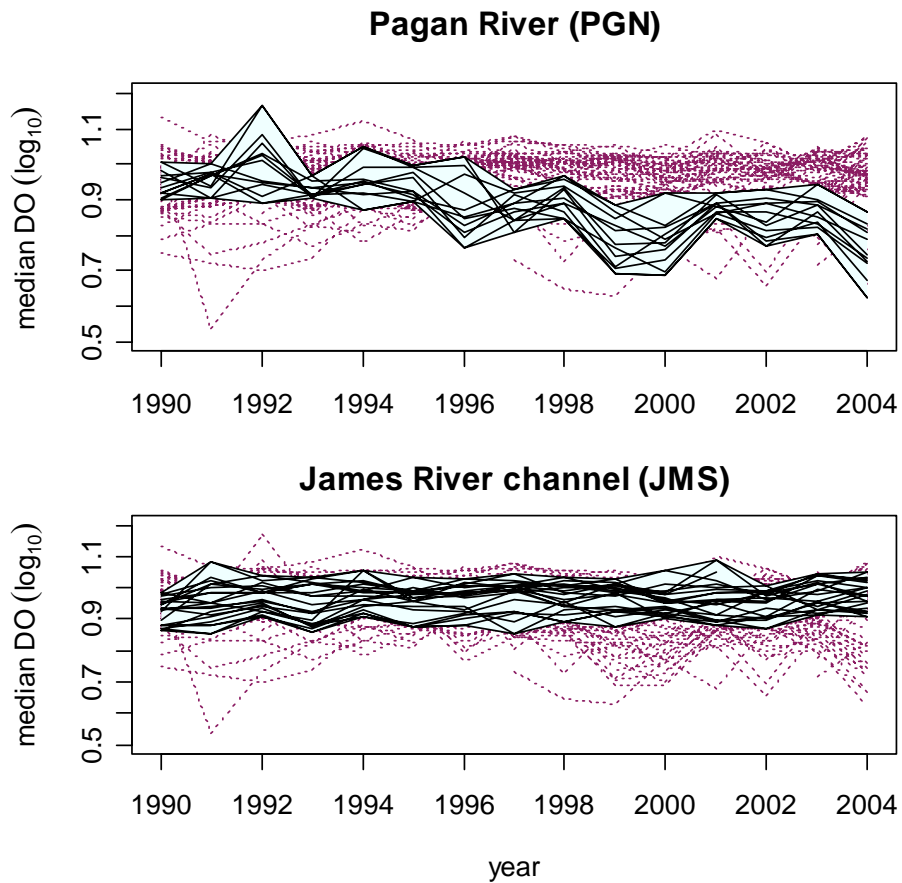
Parametric and semiparametric alternatives to a rank-based approach may have the effect of clustering based on estimated profiles that smoothed, relative to the profiles of actual measurements. Such alternatives may include functional data analysis procedures (Henderson, 2006) or incorporation of spatial or temporal autocorrelations.



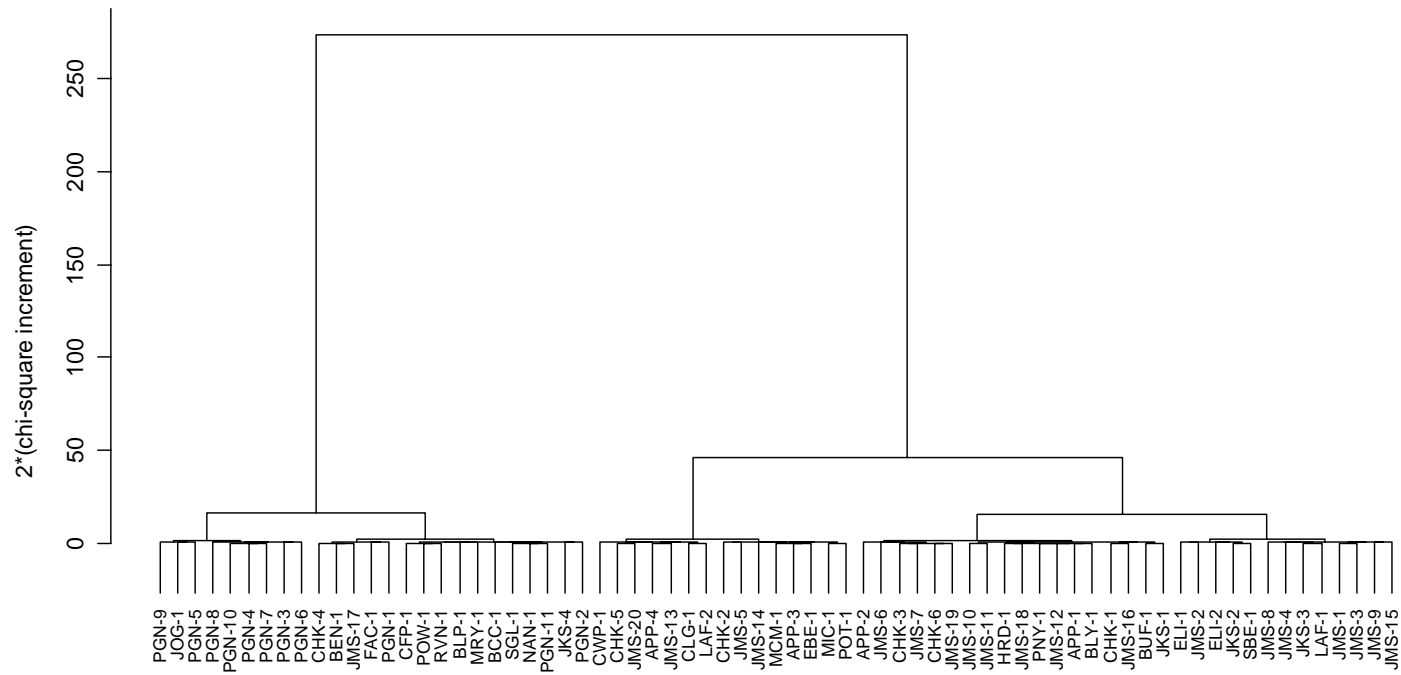
**Figure 5-1** *Locations of monitoring stations for the James River basin.* The axes represent longitude and latitude. The rectangular region at the juncture of dotted lines includes several PGN and JOG stations, and is enlarged in the figure following.



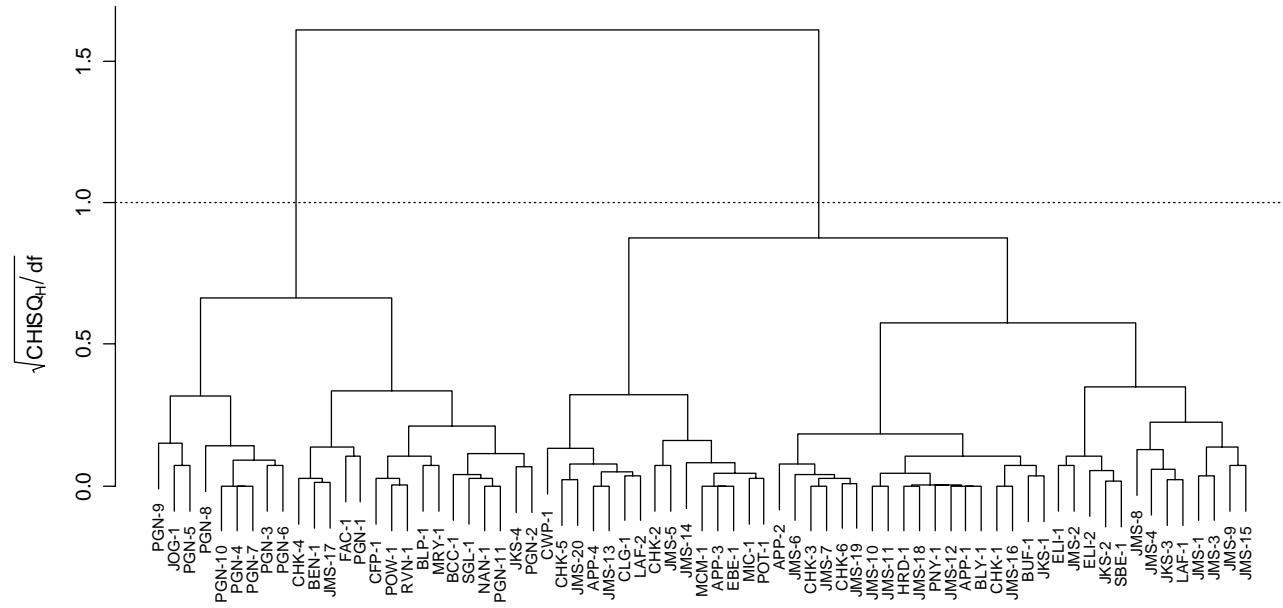
**Figure 5-2** *Detailed map of region containing Pagan River (PGN) and Jones Creek (JOG) stations.*



**Figure 5-3** Time series of dissolved oxygen ( $\text{mg}\cdot\text{L}^{-1}$ ) for stations in the Pagan River Basin (PGN) and along the James River channel. A shaded “envelope” indicates the range of concentrations for a given year. Series for other stations are displayed for comparison, using dotted lines.

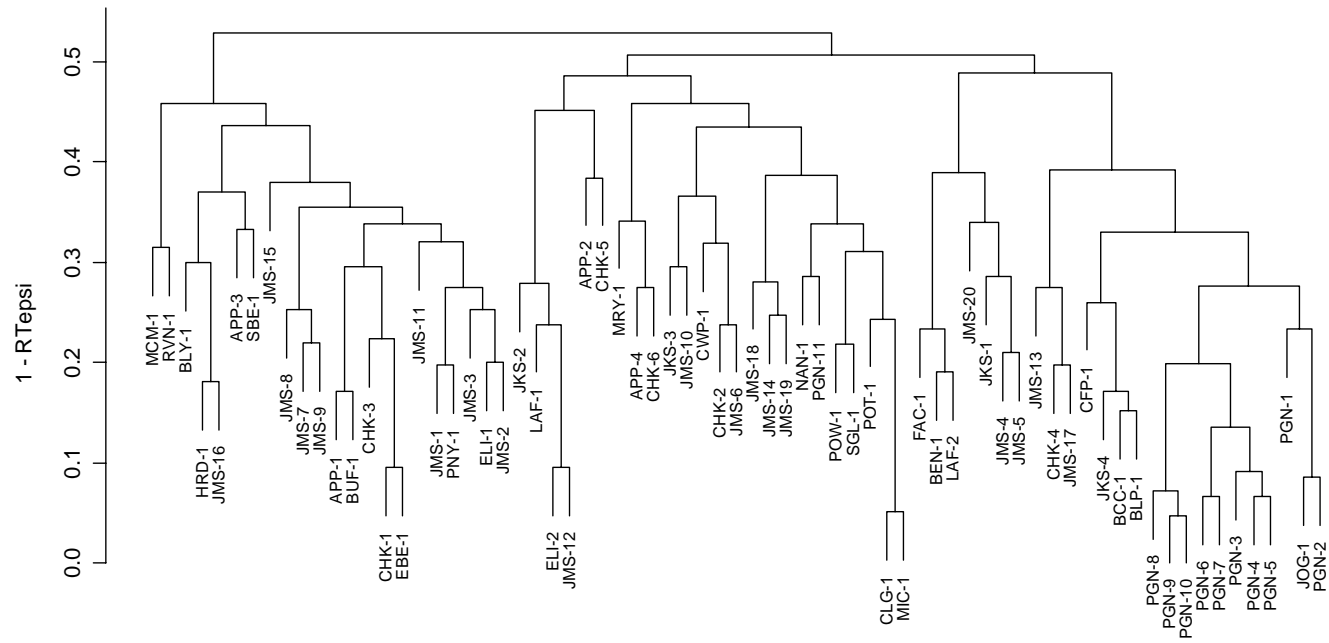


**Figure 5-4** Dendrogram based on the  $ZDiff^2$  dissimilarity criterion, Ward's linkage criterion, and the default edge lengths.

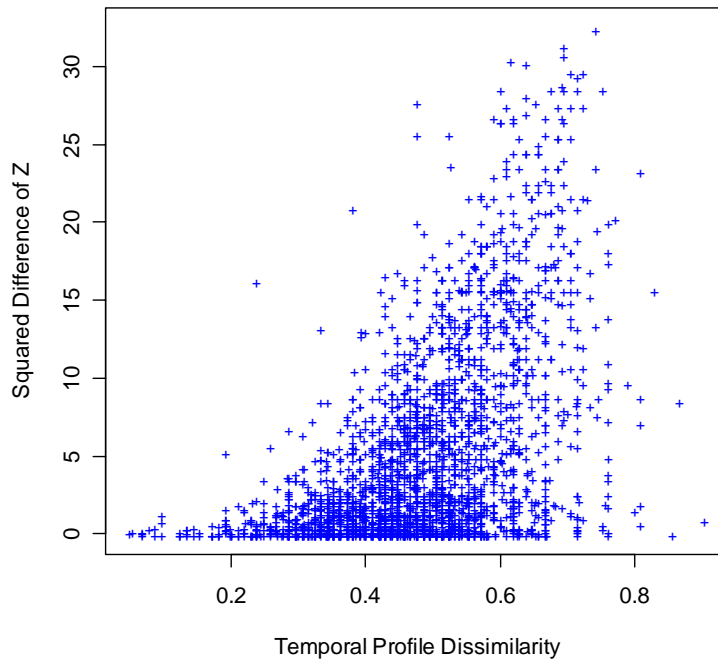


**Figure 5-5** Similar to Figure 5-4 but with the transformed edge lengths. The value of 1 (dotted line) is suggested as a reference value as discussed in the text.

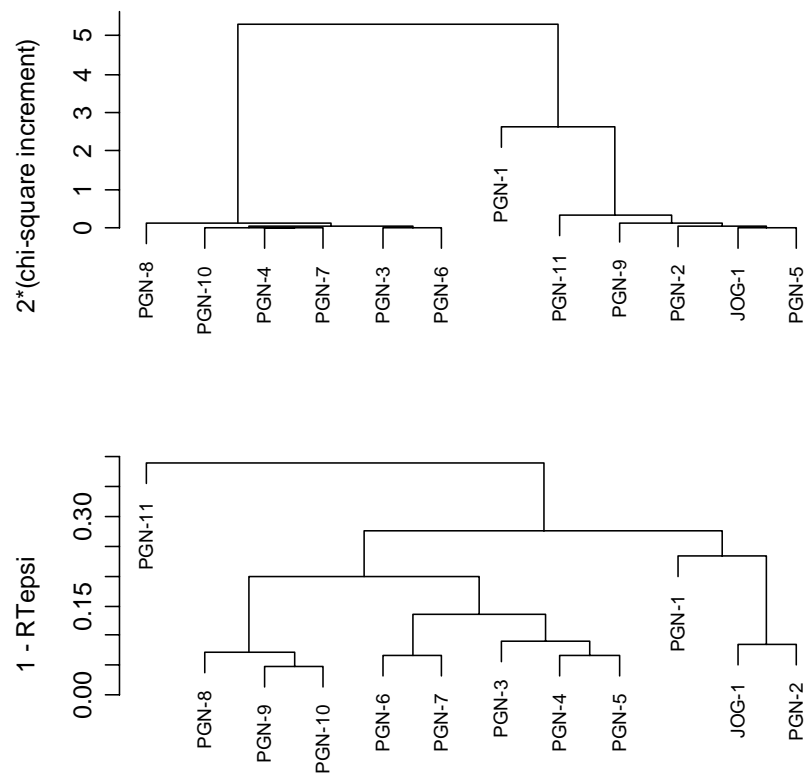




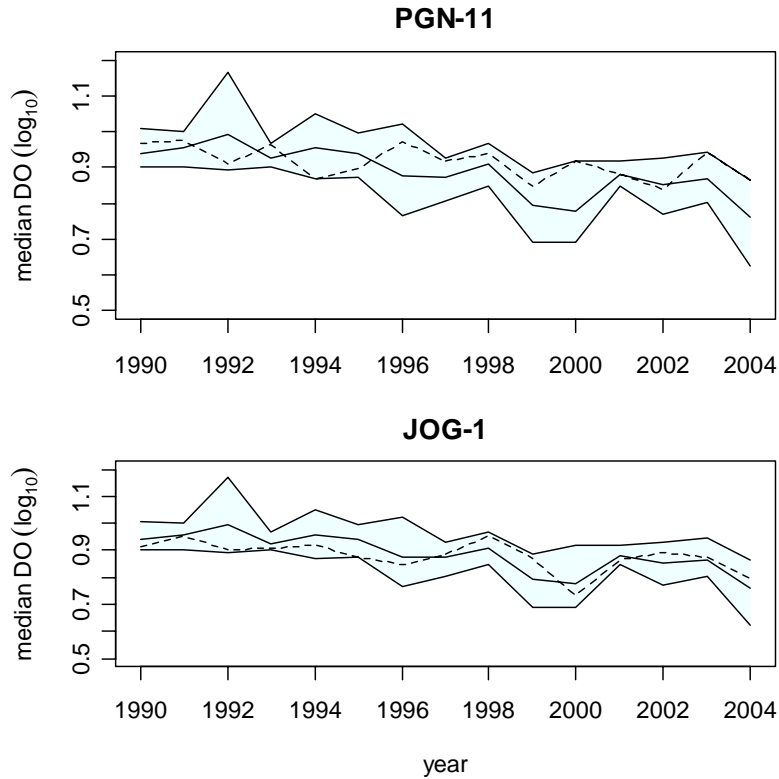
**Figure 5-6** Average linkage dendrogram based on RTepsi.



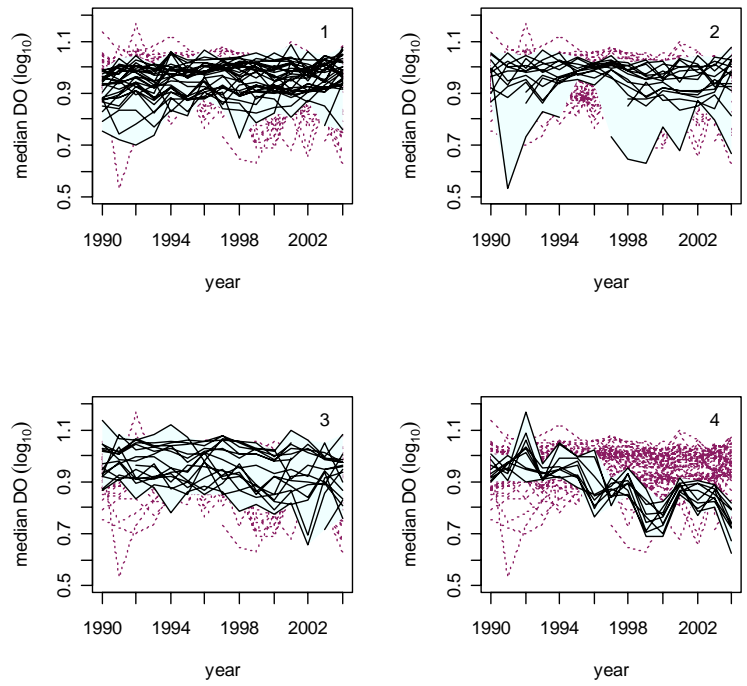
**Figure 5-7** Scatterplot comparing two distances for 71 stations. Each point plotted corresponds to a different pair of stations. The Spearman rank correlation is 0.50.



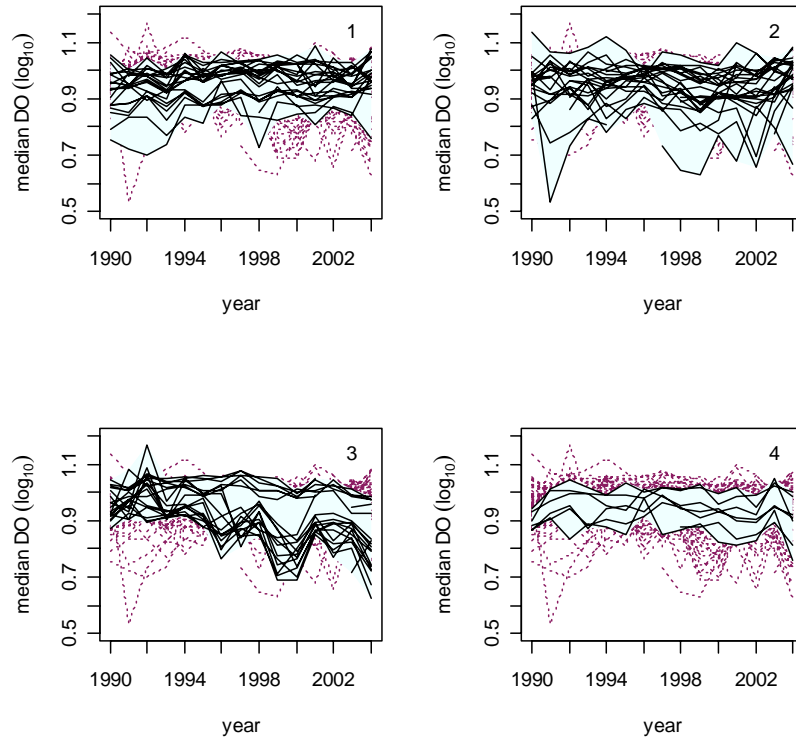
**Figure 5-8** Dendrograms for Pagan River (PGN) and Jones Creek (JOG) stations, based on the two criteria. The upper plot based is based on default R plotting heights in using Ward's method.



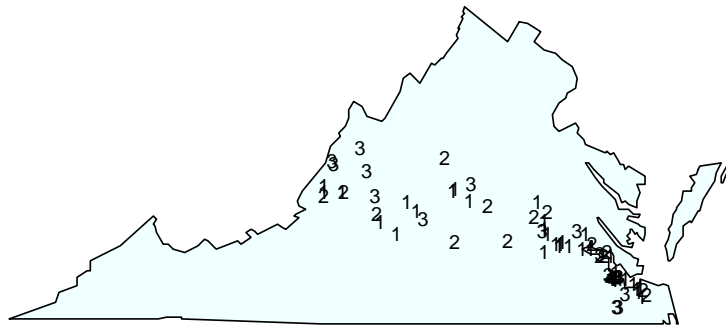
**Figure 5-9** Profiles for stations PGN-11 (upper) and JOG-1 (lower) compared to the combined set of Pagan River and Jones Creek profiles. In each plot the profile for the station of interest is represented by a dotted line. A solid line represents the annual median values for all Pagan River and Jones Creek profiles.



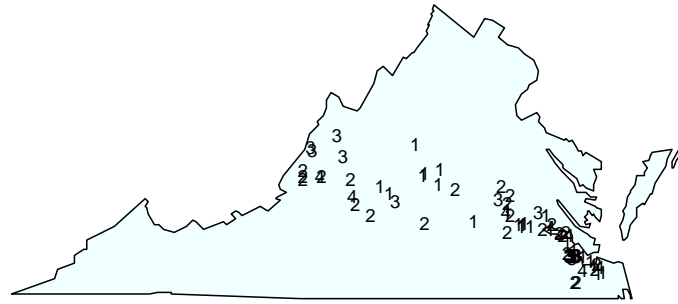
**Figure 5-10** Temporal profiles for individual stations in four clusters identified by clustering based on the  $ZDiff^2$  criterion.



**Figure 5-11** *Temporal profiles for individual stations in four clusters identified by clustering based on the RTepsi criterion.*

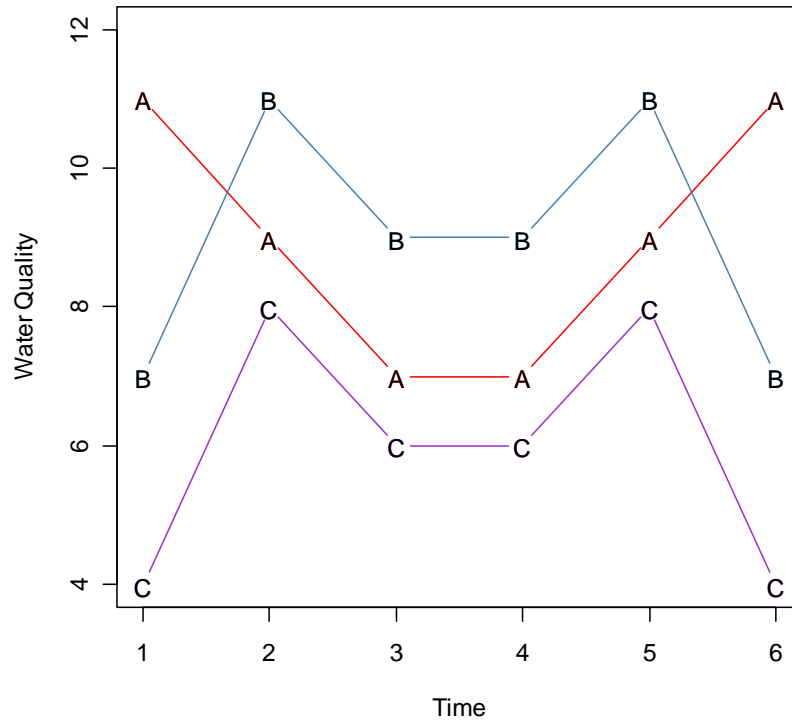


**Figure 5-12** *Coordinates of stations in 4 clusters identified by clustering based on the  $ZDiff^2$  criterion. Cluster 4 is composed of Pagan River and Jones Creek stations and is plotted separately in Figure 5-2.*

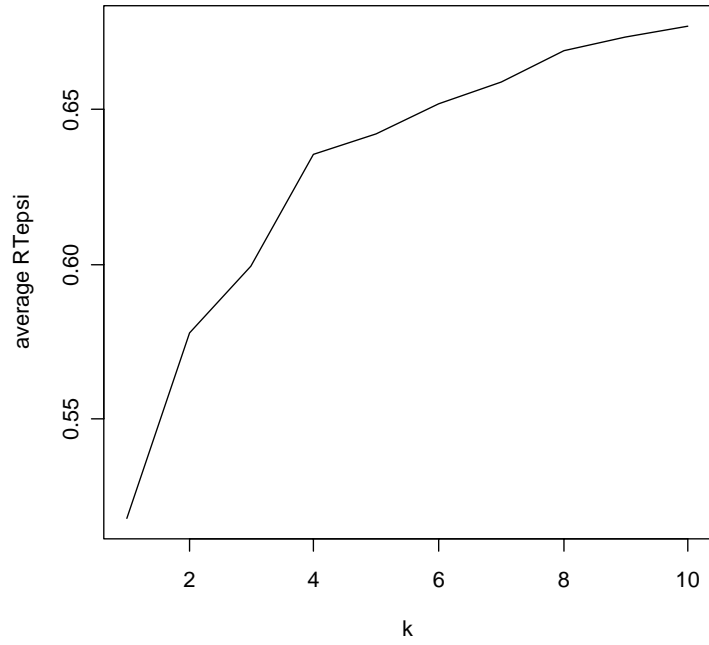


**Figure 5-13** *Coordinates of stations in 4 clusters identified by clustering based on the RTepsi criterion. Most Pagan River stations fall in C3, and are plotted separately in Figure 5-2.*





**Figure 5-14** Hypothetical temporal profiles illustrating relative advantages between the two criteria. The profiles for stations A, B, and C are indistinguishable based on Z statistics. The Z value is zero for each. However, the RTepsi is 1 comparing A and C, and 0.47 for each other pair of stations.



**Figure 5-15** Relation of averaged RTepsi to number of clusters.

## 5A APPENDIX: STATION CODES AND GEOGRAPHIC COORDINATES FOR 71 STATIONS USED IN THE ANALYSIS.

Our dataset contains the following variables.

**station** – station code as used by Virginia Department of Environmental Quality. The first 2 characters are a basin code and the next the 3 a stream code. The final 5 characters give miles between the station and the mouth of the stream.

**station.1** - compact station code generated for labeling plots. The first 3 letters are the stream code extracted from `station`. The numeric code represents separate indexing of the stations included in our analyses, when sorted according to `station` (hence sorted on mileage from mouth).

**latitude, longitude** – coordinates of the station.

**years data** – number of years with one or more measurements, out of a maximum of 15.

**Table 5A-1** James River monitoring stations.

Station					Station					Station				
St	at	ion	g	rs	St	at	ion	g	rs	St	at	ion	g	rs
o	u	d	e	a	o	u	d	e	a	o	u	d	e	a
n	.				n	.				n	.			
	l	e	d	a		l	e	d	a		l	e	d	a
				a					a					a
2-APP001.53	APP-1	37.31	-77.30	15	2-JKS023.61	JKS-2	37.79	-80.00	15	2-LAF001.15	LAF-1	36.91	-76.31	7
2-APP012.79	APP-2	37.23	-77.42	15	2-JKS030.65	JKS-3	37.84	-79.99	13	2-LAF003.83	LAF-2	36.89	-76.28	7
2-APP050.23	APP-3	37.35	-77.85	15	2-JKS058.60	JKS-4	38.04	-79.88	15	2-MCM005.12	MCM-1	38.10	-78.59	15
2-APP118.04	APP-4	37.33	-78.47	15	2-JMS005.72	JMS-1	36.95	-76.39	15	2-MIC000.03	MIC-1	37.21	-76.74	13
2-BCC004.71	BCC-1	38.07	-79.90	15	2-JMS013.10	JMS-2	36.99	-76.48	15	2-MRY014.78	MRY-1	37.75	-79.39	15
2-BEN001.42	BEN-1	36.86	-76.48	15	2-JMS021.04	JMS-3	37.06	-76.59	15	2-NAN019.14	NAN-1	36.74	-76.58	15
2-BLP000.79	BLP-1	38.20	-79.57	15	2-JMS032.59	JMS-4	37.20	-76.65	15	2-PGN000.00	PGN-1	37.01	-76.57	15
2-BLY000.65	BLY-1	37.29	-77.26	15	2-JMS042.92	JMS-5	37.20	-76.78	15	2-PGN000.80	PGN-2	37.00	-76.57	15
2-BUF002.10	BUF-1	37.61	-78.92	15	2-JMS055.94	JMS-6	37.27	-76.99	15	2-PGN001.19	PGN-3	37.00	-76.58	15
2-CFP004.67	CFP-1	37.99	-79.49	15	2-JMS069.08	JMS-7	37.30	-77.13	15	2-PGN002.58	PGN-4	37.00	-76.61	15
2-CHK002.17	CHK-1	37.26	-76.88	15	2-JMS074.44	JMS-8	37.32	-77.22	15	2-PGN003.57	PGN-5	36.99	-76.62	15
2-CHK006.14	CHK-2	37.31	-76.87	15	2-JMS075.04	JMS-9	37.31	-77.23	15	2-PGN004.57	PGN-6	36.98	-76.62	15
2-CHK023.64	CHK-3	37.40	-76.94	15	2-JMS099.30	JMS-10	37.40	-77.39	15	2-PGN005.46	PGN-7	36.99	-76.63	15
2-CHK032.77	CHK-4	37.43	-77.04	14	2-JMS104.16	JMS-11	37.45	-77.42	15	2-PGN006.65	PGN-8	36.99	-76.65	15
2-CHK062.57	CHK-5	37.60	-77.38	13	2-JMS110.30	JMS-12	37.53	-77.43	15	2-PGN007.44	PGN-9	37.00	-76.65	15
2-CHK076.59	CHK-6	37.70	-77.51	14	2-JMS117.35	JMS-13	37.56	-77.54	15	2-PGN008.42	PGN-10	37.01	-76.66	15
2-CLG000.23	CLG-1	37.23	-76.69	13	2-JMS157.28	JMS-14	37.67	-78.09	15	2-PGN010.07	PGN-11	37.02	-76.67	15
2-CWP002.58	CWP-1	37.79	-79.76	15	2-JMS176.63	JMS-15	37.71	-78.30	15	2-PNY005.29	PNY-1	37.70	-79.03	14
2-EBE002.98	EBE-1	36.84	-76.24	7	2-JMS189.31	JMS-16	37.80	-78.49	15	2-POT000.12	POT-1	37.75	-80.00	15
2-ELI002.00	ELI-1	36.90	-76.34	15	2-JMS229.14	JMS-17	37.54	-78.83	14	2-POW000.60	POW-1	37.22	-76.78	13
2-ELI004.79	ELI-2	36.87	-76.33	7	2-JMS258.54	JMS-18	37.41	-79.15	15	2-RVN015.97	RVN-1	37.86	-78.27	15
2-FAC000.85	FAC-1	37.44	-77.44	15	2-JMS275.75	JMS-19	37.51	-79.33	15	2-SBE001.53	SBE-1	36.83	-76.29	13
2-HRD011.57	HRD-1	37.81	-78.46	15	2-JMS282.28	JMS-20	37.59	-79.38	15	2-SGL001.00	SGL-1	36.74	-76.56	15
2-JKS000.38	JKS-1	37.79	-79.78	15	2-JOG000.62	JOG-1	36.99	-76.56	15					

## Chapter 6

### Discussion and Extensions

We have given three applications of cluster analysis methodology. Our interpretation of cluster analysis results can be described as relatively coarse-grained, focusing on relatively salient patterns. For example, in Chapter 3 we remark on an apparent tendency in the model-based clustering for lowland units to cluster separately from other units, particularly for datasets representing multiple physiographic regions. A more fine-grained interpretation could involve evaluation of additional biological and environmental information, along with close scrutiny of the clustering results.

Some methodological extensions that we suggest are as follows.

- The model-based approach with class semiparametric regressions may be enhanced with features for handling interactions, and more refined modeling of variances, particularly accounting for mean-variance relationships.
- Alternative spatial constraints may be considered, in addition to our approach based on grouping stations into geographic clustering units.
- The model-based procedures should be compared to alternative model based procedures (e.g., Lipkovich, 2002), perhaps with an expanded number of datasets.
- Stability of cluster analysis results may be evaluated using resampling, subsampling, or cross-validation, used to compare models that differ in the assumed number of classes and other features. We have noted that our methodology related to the label-switching problem may be useful with such alternative procedures.

- Alternative priors may be considered in Bayesian modeling, including objective Bayesian procedures or use of hierarchical priors for class regression parameters.
- We provide suggestive results in Chapter 3, related to selection of the number of clusters based on ECCP. We suggest study of the statistical properties of the approach.

## References

- Akima, H. (1978). A method of bivariate interpolation and smooth surface fitting for irregularly distributed data points. *ACM Transactions on Mathematical Software* 4:148-164.
- Akima, H. (1996). Algorithm 761: scattered-data surface fitting that has the accuracy of a cubic polynomial. *ACM Transactions on Mathematical Software*, 22:362-371.
- Bailey, R.G. (1996). *Ecosystem Geography*. Springer: New York.
- Banfield, J.D., and Raftery, A.E. (1993). Model-based Gaussian and non-Gaussian clustering. *Biometrics* 49:803-821.
- Bar-Joseph, Z., Demaine, E.D., Gifford, G.K., and Jaakkola, T. (2001) Fast optimal leaf ordering for hierarchical clustering. *Bioinformatics*, Vol. 17 Suppl. 1, pp. 22-29.
- Barbour, M.T., Gerritsen, B.D., Snyder, B.D., and Stribling, J.B. (1999). Rapid bioassessment for use in streams and wadeable rivers: periphyton, benthic macroinvertebrates, and fish. EPA 841-B-99-002. U.S. Environmental Protection Agency.
- Bensmail, H., Celeux, G., Raftery, A., and Robert, C. (1997). Inference in model-based clustering. *Statistics and Computing* 7:1-10.
- Billingsley, P. (1995). *Probability and Measure*. (3rd edition.) Wiley, New York.
- Boone, E.L., Ye, K., and Smith, E.P. (2005). Assessment of two approximation methods for computing posterior model probabilities. *Computational Statistics and Data Analysis*. 48:221-234.

- Bryant, P. G., and Williamson, J.A. (1978). Asymptotic behaviour of classification maximum likelihood estimates. *Biometrika* 65:273-281.
- Buchta, C., and Hahsler, M. (2005). *cba: Clustering for Business Analytics*. [Available with manual from the web site of the Comprehensive R archive network (CRAN) <http://cran.r-project.org/>]
- Celeux, G., Hurn, M., and Robert, C. (2000). Computational and inferential difficulties with mixture posterior distributions. *Journal of the American Statistical Association* 95:957-970.
- Chib, S., and Greenberg, E. (1995). Understanding the Metropolis-Hastings algorithm. *The American Statistician* 49:327-335.
- Dempster, A.P., Laird, N.M., and Rubin, D.B. (1977). Maximum likelihood from incomplete data via the EM algorithm (with discussion). *Journal of the Royal Statistical Society B* 39, 1-38.
- Clarkson, D.B. (1979). Estimating the standard errors of rotated factor loadings by jack-knife. *Psychometrika*, 44, 3, 297-314.
- Cohen, J. (1960). A coefficient of agreement for nominal scales. *Educational and Psychological Measurement* 20:37-46.
- Denison, D.G.T., Holmes, C.C., Mallick, B.K., and Smith, A.F.M. (2002). *Bayesian Methods for Nonlinear Classification and Regression*. Wiley, New York.
- DeSarbo, W.S., and Cron, W.L. (1988). A maximum likelihood methodology for clusterwise linear regression. *Journal of Classification* 5:249-282.



- Diebolt, J., and Robert, C.P. (1994). Estimation of finite mixture distributions by Bayesian sampling. *Journal of the Royal Statistical Society B* 56:363-375.
- Dudoit, S., and Fridlyand, J. (2002). A prediction-based method for estimating the number of clusters in a dataset. *Genome Biology* 3:0036.1 - 0036.21.
- Eilers, P.H.C. and Marx, B.D. (1996). Flexible smoothing using *B*-splines and penalized likelihood. (With comments and rejoinder). *Statistical Science* 11:89-121.
- Eubank, R.L. (1988). *Spline Smoothing and Nonparametric Regression*. Marcel Dekker, New York.
- Everitt, B. (1993). *Cluster Analysis*. (3rd edition) Edward Arnold, London.
- Fahrmeir, L., and Tutz, G. (2001). *Multivariate Statistical Modelling Based on Generalized Linear Models*. Springer, New York.
- Fan, J., and Gijbels, I. (1996). *Local Polynomial Smoothing and its Applications*. Chapman and Hall, London.
- Farrar, D.B. (2006). *Approaches to the Label-Switching Problem of Classification, Based on Partition-Space Relabeling and Label-Invariant Visualization*. Technical Report 06-7, Virginia Tech Department of Statistics.
- Fraley, C., and Raftery, A.E. (1998). How many clusters? Which clustering method? Answers via model-based cluster analysis. *The Computer Journal* 41:578-588.
- Fraley, C., and Raftery, A.E. (2002). Model-based clustering, discriminant analysis, density estimation. *Journal of the American Statistical Association* 97:611-631.

- Fraley, C., and Raftery, A.E. (2005). MCLUST: Software for Model-Based Clustering, Density Estimation and Discriminant Analysis. Technical Report 415, Dept. Statistics, U. Washington. [www.stat.washington.edu/mclust](http://www.stat.washington.edu/mclust)
- Früwirth-Schnatter, S. (2001). Markov chain Monte Carlo estimation of classical and dynamic switching and mixture models. *J. American Statistical Assoc.* 96:194-209.
- Gelman, A., and Rubin, D.B. (1992). Inference from iterative simulation using multiple sequences. *Statistical Science* 7:457-511.
- Gelman, A., Carlin, J.B., Stern, H.S., and Rubin, D.B. (2003). *Bayesian Data Analysis*. (2nd edition) Chapman and Hall, New York.
- Genz, A., Bretz, F., and Hothorn, T. (2005). *mvtnorm: Multivariate Normal and T Distribution*.
- Gilbert, R.O. (1987). *Statistical Methods for Environmental Pollution Monitoring*. Van Nostrand Reinhold, New York.
- Gilks, W.R., Richardson, S., and Spiegelhalter, D.J. (1996). Introducing Markov chain Monte Carlo. In Gilks, W.R., Richardson, S., and Spiegelhalter, D.J. (eds.) *Markov Chain Monte Carlo in Practice*. Chapman and Hall, New York.
- Gordon, A.D. (1999). *Classification*. (2nd edition) Chapman & Hall/CRC, New York.
- Govaert, G., and Nadif M. (1996). Comparison of the mixture and the classification maximum likelihood in cluster analysis with binary data. *Computational Statistics and Data Analysis*. 23:65-81.

- Gower, J.C. (1996). Some distance properties of latent root and vector methods used in multivariate analysis. *Biometrika* 53:325-338.
- Harville, D.A. (1997). *Matrix Algebra from a Statistician's Perspective*. Springer, New York.
- Hastie, T. (2005). The gam package. [Available on the web site of the Comprehensive R archive network <http://cran.r-project.org/>]
- Hastie, T. and Tibshirani, R. (1990). *Generalized Additive Models*. Chapman and Hall, New York.
- Helsel, D.R., and Hirsch, R.M. (1992). *Statistical Methods in Water Resources*. Elsevier, New York.
- Henderson, B. (2006). Exploring between site differences in water quality trends: a functional data analysis approach. *Environmetrics* 17:65-80.
- Hubert, L., and Arabie, P. (1985). Comparing partitions. *J. Classification* 2:193-218.
- Ichikawa, M. and Konishi, S. (1995). Application of the bootstrap methods in factor analysis. *Psychometrika*, 60:77-93.
- Kass, R.E. (1993). Bayes factors in practice. *The Statistician* 42:551-560.
- Kass, R.E., and Raftery, A.E. (1995). Bayes factors. *Journal of the American Statistical Association* 90:773-801.
- Hubert, L., and Arabie, P. (1985). Comparing partitions. *J. Classification* 2:193-218.
- Kendall, M.G. (1962). *Rank Correlation Methods*. (3rd edition) Hafner Publishing, New York.

- King, R.S., Baker, M.E., Whigham, D.F., Weller, D.E., Jordan, T.E., Kazyak, P.F., and Hurd, M.K. (2005). Spatial considerations for linking watershed land cover to ecological indicators in streams. *Ecological Applications* 15:137-153.
- Lamon, E.C. and Clyde, M. (2000). Accounting for model uncertainty in prediction of chlorophyll a in Lake Okeechobee. *Journal of Agricultural and Biological Statistics* 5:297-322.
- Lamon, E.C., and Stow, C.A. (2004). Bayesian methods for regional-scale eutrophication models. *Water Research* 38:2764-2774.
- Lange, K. (1999). *Numerical Analysis for Statisticians*. Springer, New York.
- Larsen, D.P., Dudley, D.R., and Hughes, R.M. (1988). A regional approach for assessing attainable surface water quality: An Ohio case study. *Journal of Soil and Water Conservation*. 43:171-176.
- Lettenmaier, D.P. (1976). Detection of trends in water quality data from records with dependent observations. *Water Quality Research* 12:1037-1046.
- Lewis, S.M., and Raftery, A.E. (1997). Estimating Bayes factors via posterior simulation with the Laplace-Metropolis estimator. *Journal of the American Statistical Association* 92:648-655.
- Lipkovich, I.A. (2002). *Bayesian Model Averaging and Variable Selection in Multivariate Ecological Models*. Dissertation, Virginia Polytechnic Inst. and State. University.
- Madigan, D., and York, J. (1995). Bayesian graphical models for discrete data. *International Statistical Review*. 63:215-232.

- MacLachlan, G., and Peel, D. (2000). *Finite Mixture Models*. Wiley, New York.
- MacLachlan, G., Peel, D., Basford, K.E., and Adams, P. (2000). The EMMIX software for the fitting of mixtures of normal and *t*-components. *Journal of Statistical Software* 4: 1-14.
- Marriott, F.H.C. (1975). Separating mixtures of normal distributions. *Biometrics* 31:767-769.
- McMahon, G., Gregonis, S.M., Waltman, S.W., Omernik, J.M., Thorson, T.D., Feeouf, J.A., Rorick, A.H., and Keys, J.E. (2001). Developing a spatial framework of common ecological regions for the coterminous United States. *Environmental Management* 28:293-316.
- Mardia, K.V., Kent, J.T., and Bibby, J.M. (1979). *Multivariate Analysis*. Academic Press, New York.
- McLachlan, G., and Peel, D. (2000). *Finite Mixture Models*. Wiley, New York.
- Mercurio, G., Chaillou, J.C., and Roth, N.E. (1999). *Guide to using 1995-1997 Maryland Biological Stream Survey Data*. Versar Inc.
- Millard, S.P., and Neerchal, N.K. (2000). *Environmental Statistics with S-Plus*. CRC Press, New York.
- Monti, S., Tamayo, P., Mesirov, J., and Golub, T. (2003). Consensus clustering: a resampling-based method for class discovery and visualization of gene expression microarray data. *Machine Learning* 52:91-118.

- Morgan, R.P., and Cushman, S.F. (2005). Urbanization effects on stream fish assemblages in Maryland, USA. *Journal of the North American Benthological Association*. 24:643-355.
- Norton, S.B. (1999). Using Biological Monitoring Data to Distinguish among Types of Stress in Streams of the Eastern Corn Belt Plains Ecoregion. Dissertation, George Mason University, Fairfax VA.
- O'Hagan, A. (1994). Kendall's Advanced Theory of Statistics. Volume 2B: Bayesian Inference. Edward Arnold, Cambridge.
- Omernik, J.M. (1987). Ecoregions of the coterminous United States. *Annals of the Association of American Geographers*. 77:118-125.
- Omernik, J.M., and Bailey, R.G. (1997). Distinguishing between watersheds and ecoregions. *Journal of the American Water Resources Association*. 33:935-948.
- Plummer, M., Best, N., Cowles, K., and Vines, K. (2005). The `coda` Package. [Available on the web site of the Comprehensive R Archive Network <http://cran.r-project.org/>]
- Poff, N. L. and Allan, J.D. (1995). Functional Organization of Stream Fish Assemblages in Relation to Hydrologic Variability. *Ecology* 76:606-627.
- Qin, L., and Self, S.G. (2006). The clustering of regression models method with applications in gene expression data. *Biometrics* 62:526-533.
- R Development Core Team (2006). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, <http://www.R-project.org>.

- Rand, W.M. (1971). Objective criteria for the evaluation of clustering methods. *J. Amer. Statist. Assoc.* 66:846-850.
- Rao, C.R. (1973). *Linear Statistical Inference and Its Applications*. (2nd edition). Wiley, New York.
- Richardson, S, and Green, P.J. (1997). On Bayesian analysis of mixtures with an unknown number of components (with discussion). *Journal of the Royal Statistical Society B* 59, 731-792.
- Robert, C.P. (2001). *The Bayesian Choice*. (2nd ed.) Springer, New York.
- Robert, C.P, and Casella, G. (2004). *Monte Carlo statistical methods*. Springer, New York.
- Roche, D.M., and Dai, J. (2003). Sampling and subsampling for cluster analysis in data mining with applications to sky survey data. *Data Mining and Knowledge Discovery* 7:215-232.
- Roeder, K., and Wasserman, L. (1997). Practical Bayesian density estimation using mixtures of normals. *Journal of the American Statistical Association* 92:894-902.
- Romesburg, H.C. (1984). *Cluster Analysis for Researchers*. Wadsworth Inc., London.
- Rossi, P.E., Allenby, G.M., and McCulloch, R. (2005). *Bayesian Marketing and Statistics*. Wiley, New York.
- Ruppert, D., Wand, M.P., and Carroll, R.J. (2003). *Semiparametric Regression*. Cambridge U. Press.

- Schwarz, G. (1978). Estimating the dimension of a model. *The Annals of Statistics* 6:461-464.
- Seber, G.A.F. (1984). *Multivariate Observations*. Wiley, New York.
- Smith, E.P., Rheem, S., and Holtzman, G.I. (1993). Multivariate assessment of trend in environmental variables. pp. 489-508 in G.P. Patil, C.R. Rao, and N.P. Ross (eds.), *Multivariate Environmental Statistics*. Elsevier Science, New York.
- Smith, M., and Kohn, R. (1996). Nonparametric regression using Bayesian variable selection. *J. Econometrics* 75:317-367.
- Stephens, M. (2000). Dealing with label-switching in mixture models. *Journal of the Royal Statistical Society B* 62:795-809.
- Tanner, M.A. (1996). *Tools for Statistical Inference*. Springer, New York.
- Ter Braak, C.J.F., Hoijsink, H., Akkermans, W., and Verdonschot, P.F.M. (2003). Bayesian model-based analysis for predicting macrofaunal communities. *Ecological Modeling* 160:235-248.
- Tibshirani, R., Walther, G., Botstein, D., and Brown, P. (2001). Cluster validation by prediction strength. Technical Report, Stanford University.
- Tierney, L., and Kadane, J.B. (1989). Accurate approximations for posterior moments and marginal densities. *J. American Statistical Assoc.* 81:82-86.
- USEPA Western Ecological Division. (2003). [Ecoregions].  
<http://www.epa.gov/wed/pages/ecoregions.htm>



- van Belle, G., and Hughes, J.P. (1984). Nonparametric tests for trend in water quality. *Water Quality Research* 20:127-136.
- Venables, W.N., and Ripley, B.D. (1998). *Modern Applied Statistics with S-Plus*. (2nd edition) Springer, New York.
- Vermunt, J.K., and Magidson, J. 2005. *Technical Guide for Latent GOLD 4.0: Basic and Advanced*. Statistical Innovations Inc., Belmont MA.  
<http://www.statisticalinnovations.com/products/LGtechnical.pdf>
- Viele, K., and Tong, B. (2002). Modeling with mixtures of linear regressions. *Statistics and Computing* 12:315-330.
- Virginia Department of Environmental Quality (VADEQ). (2006). [Web site providing detailed information on monitoring stations.] VA DEQ ,  
<https://www.deq.state.va.us/webapp/>
- Wasserman, L. (2000). Asymptotic inference for mixture models using data-dependent priors. *Journal of the Royal Statistical Society B* 62:159-180.
- Wedel, M., and Kamakura, W. (2000). *Market Segmentation*. (2nd ed.) Kluwer, Boston.
- Woods, A.J., Omernik, J.M., and Brown, D.D. (1999). Level III and IV ecoregions of Delaware, Maryland, Pennsylvania, Virginia, and West Virginia. From the USEPA (2003) web cite.

## Appendix I. R Functions Used in Model-Based Clustering

We display code for functions used to implement methods described in Chapter 2-4. Particular functions provide posterior sampling, relabeling, label-invariant visualization, and graphing of two-dimensional surfaces. The functions are not beta tested.

We suppose that a statistical model classifies  $n$  units into  $k$  disjoint subsets. The class-specific model has  $q$  regression coefficients including an intercept. An MCMC implementation generates a sample of size  $n_{MC}$ , possibly comprising multiple chains. (However, consistent notation has not been implemented for function arguments and values.)

We store the results from multiple chains, end-to-end in the same objects. (An alternative would have been to rely on lists of chain-specific results.) The interpretation of such an object is given separately in a tagged list, uniformly named `MCMCSPECS`, which includes the number of chains and the length of each chain (assumed equal for all chains). The latter object is an argument for posterior sampling routines, giving specifications for a simulation with possibly multiple chains, and for routines used in post-processing.

A sample of regression coefficients is represented by a length- $n_{MC}$  list, where each item is a  $q * k$  matrix of regression coefficients from one iteration. A sample of classifications is represented by a length- $n_{MC}$  list, where each item represents one classification.

Our methods generate samples of classifications. Such a sample is represented as a list of objects representing individual classifications, where each classification is represented as a list of objects representing clusters. (We may say that the sample is represented as a linked list.)

A single classification is represented by a length- $k$  list, where each item is a vector giving the unit indices for units in a particular. However, functions that generate or manipulate

*single* classifications rely on a more conventional representation of a classification, as a length- $n$  vector of class indices. Translation between the two representations for single classifications is handled by `asClusterList` and `asClass`.

Other unknowns may be stored in  $n_{MC} * k$  matrices, or in length- $n_{MC}$  vectors, according as values are or are not class-specific. Burn-ins are never deleted from objects that store posterior sampling results.

Our relabeling approach does not modify objects in the posterior sample, but instead generates additional “alignment” objects, which give relabeling results and can be used as arguments for post-processing functions. Our alignment object is a tagged list with two items: `$ispermuted` is a length- $n_{MC}$  vector of logicals, with values TRUE or FALSE according as relabeling does or does not permute the class indices for a given iteration, relative to original class indices (the memory locations). The value may be missing if an iteration belongs to a burn-in. A second item gives the index permutation in case of more than 2 classes (or is null in case of 2 classes, in which case the permutation is implicit). Label permutations are given by an  $n_{MC} * k$  matrix giving the ordering of indices based on relabeling. The values in a row are missing if the corresponding value in `$ispermuted` is FALSE or missing. Otherwise the  $i$ th row gives the permutation for the  $i$ th classification. The  $j$ th column records the class index after relabeling.

```

align <- function( # generate matrix where each row is index permutation

  parttn.sample, # sample of partitions (list)
  MCMCspecs,    # list of mcmc specs including K, chains, nMCMC
  G,           # num. units
  burnin=MCMCspecs$nMCMC/2, # ignored in every chain
  chainsUse=1:MCMCspecs$chains, # chains to use (default = all chains)
  prnEvry=50,  # report to console every this many iterations
  prelim.align=NULL # optional prelim alignment object (list) $ispermuted,$algnmtx
)
{
  # Relabel by maximizing trace of alignment matrix, with prelim alignment user
  # supplied or based on eccp. Input arg 1 is a sample of partitions in list format.
  # An alignment (input or output) is represented by a tagged list:
  #   $ispermuted (logical) = T iff cluster labels permuted relative to input
  #   $algnmtx =NULL for 2 classes, else nMC*N matrix of permuted class indices

  #- permutations of 1:K excluding identity permutation

  require(gregmisc)
  perms <- permutations(K,K)[-1,]
  numperms <- nrow(perms)

  #- extract MCMC specification list into variables

  K <- MCMCspecs$K # num. classes
  chains<- MCMCspecs$chains # num. independent chains
  nMCMC <- MCMCspecs$nMCMC # num. iterations per chain

  if(is.null(chainsUse)) chainsUse <- 1:chains

  if(is.null(prelim.align)) { # prelim alignment based on cru
    cat("\nComputing ECCP\n")
    eccp <- eccpFunc(parttn.sample,burnin=burnin,MCMCspecs=MCMCspecs,
                     chainsUse=chainsUse)
    cat("\nSelecting cluster representative units\n")
    dendrostuff <- eccpDendrogram(eccp,k=K,plotdendro=F)
    cru <- dendrostuff$cru # cluster representative units
    cat("\nPrelim alignment based on CRU\n")
    prelim.align <- cru.align(parttn.sample,eccp)
  }

  prelimPermuted <- prelim.align$ispermuted # logical vector
  if(K>2) prelimAlgnm <- prelim.align$algnmtx # matrix of permutations

  ispermuted <- as.logical(NA*prelimPermuted) # initialization
  if(K>2) algnmtx <- NA*prelimAlgnm
}

```

```

cat("\nsumming incidence matrices based on prelim relabeling")

nitters <- 0 # count included over chains after burn-in
for(i in chainsUse) {
  loc1 <- (i-1)*nMCMC + burnin + 1 # first to use for chain i
  loc2 <- i*nMCMC # end chain i
  for(j in loc1:loc2) { # within chain
    nitters <- nitters + 1 # count of included partitions
    parttn.j <- asClass(parttn.sample[[j]],K=K,N=G) # input partition
    if(prelimPermuted[j]) {
      if(K==2)
      {
        Class.j <- 3 - parttn.j
      } else {
        Class.j <- prelimAlgnm[j,parttn.j]
      }
    } else Class.j <- parttn.j
    Z.j <- zofgamma(Class.j) # incidence matrix, prelim aligned
    if(nitters==1) Zsum <- Z.j else Zsum <- Zsum + Z.j
    if( j==loc1 | j==loc2 | !(j%prnEvry) ) {
      cat( "\nchain= ",i, "Input locn=",j,"count included=",nitters)
    }
  }#for(j..[within chain]
}#for(i..[chains]

cat("\nrefining alignment")

numrevised <- 0 # number of permutations revised
nitters <- 0 # count included over chains after burn-in
for(i in chainsUse) {
  loc1 <- (i-1)*nMCMC + burnin + 1 # first to use for chain i
  loc2 <- i*nMCMC # end chain i
  for(j in loc1:loc2) { # within chain
    nitters <- nitters + 1 # count of included partitions
    parttn.j <- asClass(parttn.sample[[j]],K=K,N=G) # input partition
    prepermTF <- prelimPermuted[j]
    if(prepermTF) { # get prelim permutation
      if(K==2) perml <- c(2,1) else perml <- prelimAlgnm[j,]
    } else perml <- 1:K
    if(prepermTF) {
      if(K==2)
      {
        Class.j <- 3 - parttn.j
      } else {
        Class.j <- prelimAlgnm[j,parttn.j]
      }
    } else Class.j <- parttn.j
    Z.j <- zofgamma(Class.j) # incidence matrix, prelim aligned
  }
}

```

```

align.table <- crossprod(Zsum,Z.j) # alignment table
currtrace <- sum(diag(align.table)) # trace w prelim alignment
tablesum <- sum(align.table) # sum for alignment matrix

if(K==2) { # simple condition, no storage of permutation
  if(currtrace < tablesum/2) { # better does exist
    ispermuted[j] <- !prepermTF # reverse prelim
    numrevised <- numrevised + 1
  } else ispermuted[j] <- prepermTF
} else { # (K>2) evaluate all permutations
  traceByPerm <- rep(0,numperms) # trace for each permutation (init.)
  for(k in 1:numperms)
    for(l in 1:K) traceByPerm[k] <- traceByPerm[k] + align.table[perms[k,l],l]
  if(currtrace < max(traceByPerm)) { # update incidence matrix and alignment
    numrevised <- numrevised + 1
    perm2 <- perms[which.max(traceByPerm),]
  } else perm2 <- 1:K
  perm12 <- perm2[perm1] # composition of two permutations
  if(all(perm12==1:K)) { # identity
    ispermuted[j] <- F
  } else {
    ispermuted[j] <- T
    algnmtx[j,] <- perm12
  }
} # K>2

if( j==loc1 | j==loc2 | !(j%prnEvry) )
  cat( "\nchain= ",i, "Input locn=",j,"count included=",nitters)
}
}#for(i..[chains]

if(K==2) { retlist <- list(ispermuted=ispermuted,algnmtx=NULL)
} else retlist <- list(ispermuted=ispermuted,algnmtx=algnmtx)

cat("\noutput of function align:\n"); print(str(retlist))

cat("\nfraction revised re prelim alignment = ", numrevised / nitters,"\n" )

return(retlist) } #--[ end fn. defn. ]-----

```

```

asClass <- function(clusterList, K=NULL,N=NULL)
{
# Convert partition from list to vector representation

#-- determine num. clusters and num. units, if not in optional argument
if(is.null(K)) K <- length(clusterList) # num. clusters
if(is.null(N)) { N <- 0 ; for(k in 1:K) N <- N + length(clusterList[[k]]) }
Class <- rep(NA,N);
for(k in 1:K) Class[clusterList[[k]]] <- k
return(Class)
} #- end def. 'asClass' -----

asClusterList<-function(
Class,      # vector of class indices (N*1)
K=NULL     # number of classes (if null then read from argument 1)
)
{
# Convert partition from vector to list representation
# Input arg1 is a vector of class indices for units.
# I assume no gaps in indexing.
# Output is a list of vectors where kth, k=1,...,K, is the
# indices of units in the kth class.
# D. Farrar, 4/2006

if(is.null(K)) K <- max(Class)
N <- length(Class)
outlist <- as.list(rep(NA,K))
for (k in 1:K) {
  result.k <- (1:N)[Class==k]
  outlist[[k]] <- result.k
}
return(outlist)
} #- end def 'asClusterList' -----

```

```

choosereps <- function( # Choose representative of each cluster from eccp's
  eccp, # co-cluster probs (consensus) (G*G symmetric) row names required
  Class # classification as vector of cluster indice (G*1)
)
{
  # obsolete approach
  # label invariant choice of a representative from each cluster
  # From each cluster choose the unit with lowest uncertainty
  # in terms of clustering with other units.

  # cat("\n(choosereps())\n");
  # print(Class);
  # print(eccp[1:5,1:5]);

  Class <- asClusterList(Class); # reformat as list of vectors
  K <- length(Class)
  G <- nrow(eccp)
  u <- eccp*(1-eccp) # uncertainty (G*G symmetric)
  diag(u)<- NA ;
  reps <- rep(NA,K); # representative unit for each cluster (K*1)
  meanuncer <- rep(NA,G); # mean uncertainty for each unit (G*1)
  names(meanuncer)<-rownames(eccp);
  for(g in 1:G) meanuncer[g] <- mean(u[g,-g]);
  for(k in 1:K) {
    Cl.k <- Class[[k]];
    if(length(Cl.k)==1) {
      reps[k] <- Cl.k
    } else {
      t <- rep(NA,G);
      t[Cl.k]<-meanuncer[Cl.k];
      reps[k]<-which.min(t);
    }
  }
  }#for(k..
  return(reps)
}#--- end fn. defn. -----

```



```

clusterLM <- function( # selected regression results by cluster

  X,Y,                                # data (X matrix no one's, Y vector)
  Class,                               # partition in matrix form
  beta.fails=function(x) any(is.na(x)), # failure condition for beta
  locn.model=F                         # T then ignore X and compute means
)
{
  # D. Farrar 07/2006
  # generate a list of linear model esults
  # input is data and partition in matrix form
  # output is list of linear model results by cluster
  # cat("\nenter: clusterLM\n")

  K <- max(Class)

  #- initializations

  dgnList <- XPXinv <- vector(K,mode="list") # init list cluster objections
  SSR <- rep(NA,K)                          # SS residuals (K*1, init)

  if(locn.model) {
    q <- 1
  } else {
    q <- ncol(X)+1
    X <- as.matrix(X)
  }
  beta <- matrix(NA,q,K)                    # rows params, cols class (q*K,init)
  colnames(beta) <- paste("Comp",1:K)
  if(!locn.model) rownames(beta) <- c("Interc",colnames(X))

  clistLoc <- asClusterList(Class) # list of cluster index subsets

  fails <- FALSE

  for (k in 1:K) {                          # generate beta for each component

    if(k==1 || !fails) {

      Cl.k <- clistLoc[[k]] # station indices for cluster
      n.k <- length(Cl.k)  # count of units
      Y.k <- Y[Cl.k]       # response values
      dgn.k <- rep(1,n.k)
      beta.k<- SSR.k <- XPXinv.k <- NA

      if(locn.model) {

        XPXinv.k<- 1/n.k
        beta.k <- mean(Y.k)
        SSR.k <- sum((Y.k - beta.k)^2)

      } else {

        X.k <- X[Cl.k,]    # regressors
        lm.k <- lm(Y.k~X.k) # regression fit
        beta.k <- lm.k$coefficients
        fails <- beta.fails(beta.k)

        if(!fails) {

          SSR.k <-sum(lm.k$residuals^2)
          dgn.k <- cbind(dgn.k,X.k)
          colnames(dgn.k)[1] <- "Interc."
          XPXinv.k <- solve(crossprod(dgn.k))

        }

      }

    }

    if(beta.fails(beta.k)) {

```

```

    cat("\nsingular model or failure of beta constraint in clusterLM")
    fails <- T

  } else {

    beta[,k]    <- beta.k
    dgnList[[k]]<- dgn.k
    XPXinv[[k]] <- XPXinv.k
    SSR[k]      <- SSR.k

    }# if(!fails..

  }# if(k==1..
}# for(k..

if(fails) {

  cat("\nmodel rejected in clusterLM\n")
  return(list(fails=T))

} else return( list( fails=F,          # if singular or user condition
                    SSR=SSR,         # SSR by cluster (K*1)
                    dgnList=dgnList, # design matrices by cluster (length-K)
                    XPXinv=XPXinv,   # inv(X'X) by cluster (length-K list)
                    beta=beta        # beta by cluster (q*K)
                  ) )

}#-- end fn. defn. -----

```

```

computeCentroids <- function(xydata,CUndx,CUlabels) {

  # compute spatial centroid for each GCU by averaging coordinates for
  # stations belonging to a GCU
  # arg.1 (N*2) matrix x and y spatial coordinates per measurement location
  # arg.2 (N*1) vector GCU indices for use in computing GCU centroids
  # arg.3 (G*1) vector GCU labels to use as row names for output matrix
  # Longitudes should be negative for U.S.
  # (D. Farrar, 4/2006)

  centroids <- as.matrix(cbind(
    aggregate(xydata[,1],by=list(CUndx),FUN=mean)[,2],
    aggregate(xydata[,2],by=list(CUndx),FUN=mean)[,2]) )
  G <- nrow(centroids)
  if(length(CUlabels) != G) stop(
    "(computeCentroids) vector of CU labels has wrong length")
  dimnames(centroids)<-list(CUlabels,c("X","Y"))
  #-- return:
  return(centroids)
} #--- [ end fn. defn. 'computeCentroids' ] -----

```

```

cru.align <- function( # align a sample of partitions based on cru.
  parttn.sample, # sample of partitions (list, length=nMC)
  eccp           # matrix co-clustering probs (consensus matrix) (G*G)
)
{
  require(gregmisc)

  K <- length(parttn.sample[[1]]) # num. classes
  G <- nrow(eccp)

  combs <- combinations(G, K) # combinations to consider
  numcombs <- nrow(combs)
  evals <- rep(NA,numcombs)

  cat("\nchoosing cluster representative units")

  early <- F # flag for diagonal eccp submatrix

  for(i in 1:numcombs) {
    comb.i <- combs[i,]
    evals.i <- sum(eccp[comb.i,comb.i])
    if(i <= 10) cat("\n",i,comb.i)
    if(evals.i==K & i <numcombs) early <- T # submatrix is diagonal
    evals[i] <- evals.i
    if(early) break
  }

  if(i != numcombs) cru <- comb.i else cru <- combs[which.min(evals),]

  cat("\ncluster representative units=\n")
  print(cru)
  cat("\neccp for best=\n")
  print(eccp[cru,cru])

  cat("\nAligning based on cluster representative units.\n")

  nMC <- length(parttn.sample) # size of posterior sample (partitions)

  ispermuted <- rep(F,nMC) # init. output objects
  if(K > 2) algnmtx <- matrix(rep(1:K,nMC),nMC,K,byrow=T)
  for(i in 1:nMC){

    if(!(i %% 500)) cat("\n",i)
    algn.i <- rep(NA,K)
    parttn.i <- parttn.sample[[i]]
    for(k in 1:K) {
      Cl.ik <- parttn.i[[k]] # kth cluster of ith partition
      reps.ik <- intersect(Cl.ik,cru) # count of cru i in cluster k
      if(length(reps.ik)==1) algn.i[k] <- which(cru==reps.ik)
    }# for(k..
    if(!any(is.na(algn.i)) & !all(algn.i == 1:K)) {
      # alignment is successful and does permute the indices
      ispermuted[i] <- T;
      if(K > 2) algnmtx[i,] <- algn.i;
    }
  }#for(i..

  numpermuted <- sum(ispermuted);
  cat("\nNumber of clusters with labels permuted = ",numpermuted,
      "\nout of",nMC,"\n");
  if(K==2) {
    return(list(cru=cru,ispermuted=ispermuted,algnmtx=NULL))
  }else{
    for(k in 1:K) {
      cat("\ncluster ",k,"\n")
      print(table(algnmtx[,k]))
    }#for(k..
    return(list(cru=cru,ispermuted=ispermuted,algnmtx=algnmtx))
  }
} #-- end fn defn -----

```

```

eccpDendrogram <- function( # label invariant identification of clusters

  eccp,          # eccp matrix (N*N) N = num. units. (consensus mtrx)
  leaflabels=rownames(eccp), # optional leaf labels (default-eccp row names)
  cex=0.7,      # character expansion for leaf labels
  k,           # num. clusters to return
  plotdendro=T # Plot dendrogram if TRUE

)
{
  # Determines dendrogram based on eccps (consensus matrix of Monti et al.)
  # based on 1 - eccp as a dissimilarity in average linkage.
  # Clusters are optionally labeled according to representative units
  # returns average eccp for a unit with members of its cluster.

  # (D. Farrar 4/2006)

  G <- nrow(eccp)          # num. units

  codist <- as.dist(1-eccp) # base R fn. returns distance matrix
  hc <- hclust(codist, method="ave") # store ave linkage dendrogram
  if(plotdendro) {
    hcpl <- hc              # make copy and modify for plotting
    hcpl$method<-hcpl$call<-NULL # otherwise printed on dendrogram
    plot(hcpl,labels=leaflabels,main="",sub="",xlab="",cex=cex,
         ylab="Distance = 1 - ECCP")
  }

  #-- extract clusters

  class0 <- cutree(hc,k=k) # un-aligned classfn, ok if h or k NA

  #-- label based on representative units

  cru <- sort(choosereps(eccp,class0)) # choose a rep unit from each cluster
  names(cru) <- paste("Cl.",1:k,sep="") # relabel
  # cat("\nrepresentative units = ",cru,"\n")
  Class<- NA*class0          # aligned class
  for(i in 1:k) {
    currentC.i <- class0[cru[i]] # current cluster for ith rep
    Class[class0==currentC.i] <- i # replace cluster index
  }

  #-- compute average similarity of each unit with others in its cluster

  return(list(Class=Class))
} #----[ end fn. defn. ]-----

```

```

eccpFunc <- function( # estimated co-clustering probs (consensus matrix)

  parttn.sample,          # list of partitions
  MCMCspecs,             # MCMC specifications
  burnin=MCMCspecs$nMCMC/2, # burnin deleted from each chain
  chainsUse=1:MCMCspecs$chains, # chains to use
  unitLabels=NULL        # row and col names for eccp matrix

)
{
  # compute estimated co-clustering probabilities from sample of partitions
  # arg. 1 is sample of partitions in list form, possibly with multiple chains
  # (D. Farrar 6/2006)

  parttn1 <- parttn.sample[[1]]
  K <- length(parttn1)
  G <- 0
  for(k in 1:K) G <- G + length(parttn1[[k]])
  cat("\nComputing eccp matrix with burnin = ",burnin)
  cocounts <- matrix(0,G,G,dimnames=list(unitLabels,unitLabels))
  for(i in chainsUse) {
    # moved partitions to linked list after burnin
    loc1 <- (i-1)*nMCMC + burnin + 1 # locn 1 to use in chain i
    loc2 <- i*nMCMC                # end of chain i
    cat("\nchain = ",i,"list locations ",loc1," - ", loc2)
    for(j in loc1:loc2) {
      parttn.ij <- parttn.sample[[j]]
      for(k in 1:K) {
        cl.ijk <- parttn.ij[[k]]
        cocounts[cl.ijk,cl.ijk] <- cocounts[cl.ijk,cl.ijk]+ 1
      } # for(k ... [clusters within iteration]
    } # for(j ... [iterns within chain]
  }; cat("\n") # for(i ... [over chains]
  return( cocounts / (length(chainsUse)*(nMCMC-burnin)) )
} #--- [ end fn. defn. 'eccpFunc' ]-----

```

```

ecmpFunc <- function(
  parttn.sample,      # list of partitions
  MCMCspecs,          # includes $chains etc.
  alignment=NULL,     # alignment information $ispermuted, $algnmtx
  burnin=MCMCspecs$nMCMC/2, # burnin deleted from each chain
  unitLabels=NULL,    # row names for output matrix
  chainsUse=1:MCMCspecs$chains # vector chains to use
)
{
  # Computes estimated class membership probabilities (ecmp).
  # Compare to "eccpFunc"
  # D. Farrar 4/2006

  cat("\ngenerating matrix of ecmps\n");
  parttn1 <- parttn.sample[[1]];
  K <- length(parttn1);
  B <- 0;
  for(k in 1:K) B <- B + length(parttn1[[k]]);

  chains <- MCMCspecs$chains;
  nMCMC <- MCMCspecs$nMCMC;

  if(is.null(alignment)) {
    ispermuted <- rep(F,length(parttn.sample))
  } else {
    ispermuted <- alignment$ispermuted
    algnmtx <- alignment$algnmtx
  }

  ecmp <- matrix(0,B,K,dimnames=list(unitLabels,paste("Component",1:K)))

  for(i in chainsUse) { # over chains

    loc1 <- (i-1)*nMCMC + burnin + 1
    loc2 <- i*nMCMC

    cat("\nchain",i,"iterations",loc1," - ",loc2)

    for(j in loc1:loc2) { # over partitions in chain

      parttn.ij<- parttn.sample[[j]]

      if(ispermuted[j]) {

        if(K==2) perm.ij <-c(2,1) else perm.ij <- algnmtx[j,]
        for(k in 1:K) {
          cl.ijk <- parttn.ij[[k]]
          ecmp[cl.ijk,perm.ij[k]] <- ecmp[cl.ijk,perm.ij[k]] + 1
        } #for(k)
      } else {

        for(k in 1:K) {
          cl.ijk <- parttn.ij[[k]]
          ecmp[cl.ijk,k] <- ecmp[cl.ijk,k] + 1
        } #for(k)

      } #if

    } # for(j ... [within chain]
  };cat("\n\n");# for(i ... [over chains]

  ecmp <- ecmp / (length(chainsUse)*(nMCMC-burnin));
  return(ecmp) # num. units*K;
} #-- [ end fn. defn. 'ecmpFunc' ] -----

```

```

fetch.betas <- function( # retrieve betas from posterior sample into matrix
  beta.sample, # list of q*K matrices of betas (rows regrssrs/cols clustrs)
  clset, # vector indices of classes of interest
  betaset, # vector indices betas of interest
  algnmtx=NULL # each row gives permutation of component indices (n.iters*K)
  # ("alignment matrix")
)
{
  # Obsolete specification of alignment
  # Assumes a sample of regression betas in the form of a list
  # of matrices (columns for clusters, rows for regressors)
  # Creates a matrix of betas for input subsets of clusters and regressors

  niters <- length(beta.sample)
  numcl <- length(clset)
  numbeta<- length(betaset)
  outmtx <- matrix(NA,niters,numcl*numbeta)
  #-- output matrix with column names
  colnames(outmtx) <- rep("",numcl*numbeta)
  loc <- 1
  for (i in betaset) for (j in clset) {
    colnames(outmtx)[loc] <- paste("C",j,"X",i,sep="")
    loc <- loc + 1
  }; cat("\n")
  #-- fetch betas into matrix
  for(i in 1:niters) {
    beta.i <- beta.sample[[i]]
    if(!is.null(algnmtx)) beta.i <- beta.i[,algnmtx[i,]]
    loc <- 1
    for (j in betaset) {
      for (k in clset) {
        outmtx[i,loc] <- beta.i[j,k]
        loc <- loc + 1
      }#for(k..)
    }#for(j..)
  }#for(i..)
  return(outmtx)
} #-- end function defn. -----

```



```
index <- function(invec,valueset=NULL) {  
  
  # numeric indexing of categories.  
  # Input is a vector to be interpreted as categorical,  
  # say with K categories.  Output vector is of same  
  # length and indexes the categories 1,...,K.  
  # Optional 2nd argument gives the categories present in input.  
  # D. Farrar 4/2006  
  
  if(is.null(valueset)) valueset <- sort(unique(invec))  
  outvec <- apply(as.array(invec), 1, function(xx){which(valueset==xx)})  
  
  return(outvec)  
} #-----
```

```

linmixture.GCU <- function(
  MCMCspecs,      # tagged list of specifications, see first code block
  X=NULL,         # matrix of regressors not including column of 1's
  Y,              # response vector
  CU0,           # GCU code, one value per observation
  beta.fails=function(x) any(is.na(x)), # constraint on beta (T if failed)
  prnEvry=25,    # report to monitor every this many iterations
  pl=T           # T then plot log-likelihood
)
{
  # Posterior sampling with class linear model and multiple-station clustering units.

  # * GCU code may be string or numeric and may have gaps
  # * (function will index the GCU)

  require(mvtnorm) # multivariate normal lib

  maxtries<-MCMCspecs$maxtries # max tries to meet constraints
  cru      <-MCMCspecs$cru      # cluster representative units
  K        <-MCMCspecs$K        # num classes
  save.betas<-MCMCspecs$save.betas # F then discard beta sample
  chains   <-MCMCspecs$chains   # num chains
  nMCMC    <-MCMCspecs$nMCMC    # length of each chain
  statns.min<-MCMCspecs$statns.min # min stations per cluster
  do.align<-MCMCspecs$do.align  # if T then generate alignment object
  modlFM   <-MCMCspecs$modlFM   # if T then finite mixture (FM)
  mintau   <-MCMCspecs$mintau   # min mixing fraction (FM only)
  alpha    <-MCMCspecs$alpha    # Dirichlet parameter
  thinval  <-MCMCspecs$thinval  # 2 then double chains, save even
  homogvar <-MCMCspecs$homogvar # 2 then double chains, save even

  if(length(unique(CU0)) < K) stop("Fewer than ",K," GCU in input")

  #-- automated over-rides for special cases

  do.thin <- !is.null(thinval)
  if(do.thin) nMCMC <- nMCMC*thinval
  if(K == 1) {modlFM <- F; chains <- 1; cru <- NULL}
  force.cru <- !is.null(cru)
  if(!modlFM) alpha <- mintau <- NULL

  #-- counting and coding

  N <- length(Y) # num stations
  CU <- index(CU0) # GCU indices in place of input labels
  G <- max(CU) # num GCU
  CUndxSet<-1:G # vector of unique values
  CUcounts<-as.vector(table(CU)) # count station for each CU (G*1)

  #-- design matrix with column of ones

  dgn <- as.matrix(rep(1,N))
  locn.model<-is.null(X) # then fit only location differences
  if(!locn.model) dgn <- cbind(dgn,X)
  q <- ncol(dgn)

  #-- CU codes to use in matrix row and col labels

  culabels0<- sort(unique(CU0)) # input GCU set/ may have gaps if numeric
  names(CUndxSet)<- culabels0
  culabels <-paste(paste(CUndxSet, ". ",sep=""),culabels0,sep="")
  names(CUcounts)<- culabels
  classlabels <- paste("Cl.",1:K,sep="") # class labels (K*1)

  #-- matrices with named dimensions for initializations

  initN.NA <-rep(NA,N)
  initNK.NA<-matrix(NA,N,K,dimnames=list(NULL,classlabels)) # N*K, all NA
  initGK.NA<-matrix(NA,G,K,dimnames=list(culabels,classlabels)) # G*K, all NA

```

```

#-- Initialize objects where sample is stored
# * betas are stored as a list of q*K matrices

nitters.actual <- nitters.saved <- chains*nMCMC      # sample size for all chains
if(do.thin) nitters.saved <- nitters.saved / thinval

cat("\niterations to generate = \t",nitters.actual)
cat("\niterations to save = \t",nitters.saved,"\n")

logLik.sample <- rep(NA,nitters.saved)
sigma.sample  <- matrix(NA,nitters.saved,K)

if(modlFM) tau.sample <- matrix(NA,nitters.saved,K-1,
    dimnames=list(NULL,c(paste("tau.",1:(K-1),sep=""))))

parttn.sample <- vector(nitters.saved,mode="list")  # init. sample of partitions

beta.sample <- NULL

if(save.betas) beta.sample<-vector(nitters.saved,mode="list") # init. beta sample

#-- initializations for Metropolis loop

itrn <- 0          # counter for iterations (constraints not failed)
chain<- 0         # counter for chains
done <- FALSE     # flag for termination of simulation
startTime<-Sys.time() # system time at start of simulation

if(K==1) {        # one class
  regrstuff<-clusterLM(X,Y,Class=rep(1,N), # beta.fails=beta.fails,
    locn.model=locn.model)
  # SSR <- sum(regrstuff$SSR) # SS residuals, sum over clusters
}

while(!done) {   # Metropolis loop

  trynum <- 0    # counter for attempts to meet constraints
  fails  <- TRUE # flag for acceptance of update

  while(fails & (trynum<=maxtries)) { # loop until constraints met

    newchain <- !(itrn %% nMCMC) # new chain (works for counter init 0)

    if(K > 1) {

      if(newchain) { # stuff for new chain, initializations

        #-- initialize objects that can be updated within a chain

        regrstuff.0 <- regrstuff <-NULL # list regression results
        # SSR.0 <- SSR <-NULL # vector residual SS
        statnclass.0<- statnclass<-NULL # station classification
        CUclass.0 <- CUclass <-NULL # GCU classification
        nStatn.0 <- nStatn <-NULL # count stations per cluster
        nCU.0 <- nCU <-NULL # count of GCU per cluster
        rbeta.0 <- rbeta <-NULL # beta ~ posterior
        rsigma.0 <- rsigma <-NULL # sigma ~ posterior
        tau.0 <- tau <-NULL # tau ~ posterior

        lzhatx <- NULL # un-normalized log z-hat

        #-- init. classification (CUs and stations)

        CUclassL <- rpartition(N=G,K,force=cru) # random CU classfn(min 1) (init)
        CUclass.0<- CUclassL$class
        nCU.0 <- CUclassL$n # counts of CU per class

        #-- classify stations based on GCU classes,
        # * check min stations per class

```

```

statnclass.0 <- initN.NA      # classification of stations
for (i in 1:N) statnclass.0[i]<-CUclass.0[CU[i]]
nStatn.0 <- as.vector(table(statnclass.0)) # count stations per class

fails <- (min(nStatn.0) < statns.min)

cat("\nInitial partition\n")
str(CUclassL)

#-- initialize lists of cluster regression results
if(!fails) {
  regrstuff.0 <- clusterLM(X,Y,Class=statnclass.0, # beta.fail=beta.fail,
                          locn.model=locn.model)

  fails <- regrstuff.0$fails # missing encountered

  # if(!fails) SSR.0 <- sum(regrstuff.0$SSR)
}#if(!fails..

#-- specify a random cycle of units that can move
if(force.cru) canmove <- setdiff(1:G,cru) else canmove <- 1:G
proporder<- sample(canmove) # 'proposal order' for CRU
j<-1 # start at first locn. in vector
} else { #-- continuation of current chain (no new chain)-----

#-- make copies of objects that may be updated
# * acceptance is subject to constraints for these objects

regrstuff.0 <-regrstuff; statnclass.0 <-statnclass
CUclass.0 <-CUclass; nCU.0 <-nCU; nStatn.0 <-nStatn; # SSR.0 <- SSR

#-- proposal of new partition subject to constraints:
# * min. cluster size (stations)

g <- proporder[j] # propose CU from cycle
CC <- CUclass.0[g] # current cluster of CU proposed
candiC <- setdiff(1:K,CC) # candidates for new cluster
PC <- ifelse(K==2,candiC,sample(candiC,1)) # proposal cluster
fails <- ((nStatn.0[CC]-CUcounts[g]) < statns.min)

#-- acceptance / rejection

acceptmove <- F

if(!fails) { # propose partition and acceptance step
  # ratio full conditionals for acceptance
  r <- exp(lzhatx[g,PC] - lzhatx[g,CC])
  acceptmove <- ifelse(r > 1, T, ifelse(runif(1)<r, T, F))
  if(F) cat("\nok 1")
}

#-- update partition and regressions if proposal accepted

if((!fails) && acceptmove) {
  CUclass.0[g]<-PC # CU classification
  statnclass.0[CU==g]<-PC # classification of stations (N*1)
  nCU.0[PC]<-nCU.0[PC]+1 # increment GCU count, proposed cluster
  nCU.0[CC]<-nCU.0[CC]-1 # decrement GCU count, current cluster
  nstatnsmove <- CUcounts[g] # num. stations to move
  nStatn.0[PC]<-nStatn.0[PC]+nstatnsmove # incr station count (proposed clus)
  nStatn.0[CC]<-nStatn.0[CC]-nstatnsmove # decr station count (current)
  if(K==2) {
    regrstuff.0 <- clusterLM(X,Y,Class=statnclass.0, # beta.fail=beta.fail,
                            locn.model=locn.model)
  } else {
    clistStatn <- asClusterList(statnclass.0) # format classifn. as list
  }
}

```

```

Cl.P <- clistStatn[[PC]] # station indices for current cluster
regrstuff.0 <- replace.clusterLM(regrstuff.0,PC,X[Cl.P,],Y[Cl.P],
                                locn.model=locn.model)
if(!regrstuff.0$fails) {
  Cl.C <- clistStatn[[CC]]# retrieve indices
  regrstuff.0 <- replace.clusterLM(regrstuff.0,CC,X[Cl.C,],Y[Cl.C],
                                locn.model=locn.model)
}
}
fails <- regrstuff.0$fails # missing values encountered
# if(!fails) SSR.0 <- sum(regrstuff.0$SSR)
}

}# if([new chain]) {} else {..[continued chain]}
}# if(K > 1) -----

### Update objects that are updated in the same way for a new or continued
### chain. Objects subject to failure conditions.

if(K == 1) { # one-cluster model (Kludge)
  fails = F
  regrstuff.0 <- regrstuff
  # SSR.0 <- SSR
} else {
  j <- j+1 # next GCU in cycle
  if(j > length(canmove)) { # new cycle of GCU
    proporder <- sample(canmove)
    j <- 1
  }#if(j)
}#if(K)

#-- sample regression model params from full conditionals

if(!fails) { # simulate betas from multivariate normal

  # rsig2 <- SSR.0 / rchisq(1,N) # s^2~inverse chi

  if( homogvar ) {
    rsig2 <- rep( sum(regrstuff.0$SSR)/rchisq(1,N), K)
  } else {
    rsig2 <- rep(NA,K)
    for(k in 1:K) rsig2[k] <- regrstuff.0$SSR[k] / nStatn.0[k]
  }

  rsigma.0 <- sqrt( rsig2 ) # random sigma
  rbeta.0 <- matrix(NA,q,K) # rows params, cols clusters

  colnames(rbeta.0) <- paste("Cl.",1:K,sep="")

  if(!locn.model) rownames(rbeta.0) <- c("Interc",colnames(X))

  for (k in 1:K) { # generate beta for each component

    if(k==1 || !fails) {

      mub.k <- regrstuff.0$beta[,k] # mean vector
      varb.k <- rsig2[k]* regrstuff.0$XPXinv[[k]]
      if (q==1) rbeta.k<- rnorm(1, mub.k, varb.k) else
        rbeta.k<-rmvnorm(1, mub.k, varb.k)
      rbeta.k <-as.vector(rbeta.k)
      fails <- beta.fails(rbeta.k,regrstuff.0$dgnList[[k]])
      if(!fails) rbeta.0[,k]<-rbeta.k

    }#if(k==1..

  }# for(k..

  if(F) cat("\nok 3")
}# if(!fails..

```

```

#-- (mixture model) update mixing fractions subject to min value

if( (!fails) & modlFM ) {
  # * modlFM=F automatically when K=1
  tau.0 <- rDirichlet1(alpha+nCU.0)
  if(!is.null(mintau)) fails<-(min(tau.0)<mintau)
} else tau.0 <- rep(NA,K)

trynum <- trynum + 1 # num attempts to meet constraints
} # while(fails & (trynum<=maxtries)){..

if(fails) { # failure after max tries exceeded -----

  done <- TRUE
  cat("\n! Terminated : constaints not met in max attempts =",maxtries)

} else { # updates successful - save results -----

  # * updates not subject to failure conditions

  itrn <- itrn + 1 # count good iterations (init 0)
  done <- (itrn==niters.actual) # determine termination

  chain<- chain + newchain # chain of current itrn

  #-- copy objects resulting from successful updates

  rbeta <- rbeta.0; rsigma <- rsigma.0

  if(K > 1) { tau <- tau.0; regrstuff <- regrstuff.0; # SSR <- SSR.0
    statnclass <- statnclass.0; CUclass<- CUclass.0
    nCU <- nCU.0; nStatn <- nStatn.0 }

  #-- Partition in form of a list of clusters
  if(K > 1) cStatnsList <- asClusterList(CUclass,K)

  #-- log-likelihood
  # * un-normalized densities
  # * organization: if there is a failure condition, put in loop above.

  #-- contribution of each station
  lf1 <-initNK.NA; # logs of densities f_k(y_i)
  for (k in 1:K) lf1[,k]<-dnorm(Y, dgn %**% rbeta[,k], rsigma[k], log=T);

  if(K==1) {
    logLik <- sum(lf1)
  } else {
    # sum over station per CU
    lf2 <-aggregate(lf1, by=list(CU),FUN=sum)[,2:(K+1)]
    dimnames(lf2) <- dimnames(initGK.NA)
    if(modlFM) { # finite mixture model
      lzhatx <- initGK.NA # tau*f = un-normalized zhat (G*K)
      for (k in 1:K) lzhatx[,k]<-log(tau[k])+lf2[,k]
    } else { # classification model
      lzhatx <- lf2
    }
    # classification log-likelihood
    logLik <- 0
    for(k in 1:K) logLik <- logLik + sum(lzhatx[cStatnsList[[k]],k])
  }

  #-- reformat partition from vector of cluster indices (gamma)
  #-- to list of clusters

  # ---monitor simulation
  if((itrn==niters.actual)|(!(itrn%prnEvry))|((itrn %% nMCMC)<=5)){
    cat( "\n\n", chain, itrn,"/",niters.actual, "sigma =",rsigma,"\n" )
    print(t(rbeta))
    if(modlFM) cat("\ntau =",tau)
    #-- report current partition (use input CU indices not arr statnns)
  }

```

```

    cat("\nlog-likelihood",logLik,"\n");
    if(K > 1) for (k in 1:K)
      cat("\ncluster",k,":",CUndxSet[cStatnsList[[k]])]
  }

  #-- store results
  # * (if no thinning, or even iteration)

  if(!(do.thin) || !(itrn%%thinval)) {

    locsave <- itrn
    if(do.thin) locsave <- locsave / thinval
    logLik.sample[locsave] <- logLik
    sigma.sample[locsave,] <- rsigma
    if(save.betas) beta.sample[[locsave]] <- rbeta
    if (K>1) parttn.sample[[locsave]] <- cStatnsList
    if(modlFM) tau.sample[locsave,]<-tau[1:(K-1)]

  }

  } #--- stuff based on good iterations (save stuff, monitor)
}; cat("\n"); #--- end Metropolis loop -----
cat("\nnum. stations = ",N,"\nnum. GCU = ",G);

if(pl) dmyy <- logLikPlot(logLik.sample,MCMCspecs)

#-- return:
cat("\n")

return(list(logLik=logLik.sample, # vector
           sigma =sigma.sample, # vector
           beta =beta.sample, # list of q*K matrices
           parttn=parttn.sample # linked list
           ))

}#-- end fn defn -----

```

```

listOfDframes <- function(
  x,          # [N*?] data frame
  Classes,   # [K*1] partition in vector form (jth element class of unit j)
  byvar      # [N*1] unit code common to arg 1 and arg 2
)
{
  # utility. Given a data set and a partition of GCU, where
  # each GCU may include multiple stations, reformat
  # the data as a list of cluster-specific data frames.
  # I do not use this in my MCMC routines.
  # D. Farrar 5/2006

  K <- max(Classes)
  clusterList <- asClusterList(Classes)
  listDt <- vector(K,mode="list")
  cat("\nformatting data according to cluster.")
  for(j in 1:K) {
    Cl.j <- clusterList[[j]]
    cat("\n",j,"=",Cl.j)
    listDt[[j]] <- subset(x, is.element(byvar,Cl.j))
  };cat("\n")
  return(listDt)
}

logLikPlot <- function(

  x,          # vector of values to plot, possibly multiple chains
  MCMCspecs, # list includes chains and number / chain
  plotevery=1, # thinning, e.g., 5 for plot every 5th (20%)
  main="",
  ylim=range(x)
)
{
  # log-likelihood plotted against position in chain
  # all chains should be in one vector, num. &length of chains in 2nd argument

  chains <- MCMCspecs$chains
  nMCMC <- MCMCspecs$nMCMC
  niters <- length(x)

  plot(c(1,chains*nMCMC),range(x), type="n", axes=F,
        main=main, ylim=ylim,
        xlab="chain",ylab="log-likelihood",col="blue")
  axis(2) # ordinate
  axis(1, at=(0:chains)*nMCMC, labels=F) # ticks
  axis(1, at=((1:chains)-1/2)*nMCMC, tick=F, labels=1:chains) # chain nums.

  for(i in 1:chains) {
    loc1 <- (i-1)*nMCMC+1
    loc2 <- i*nMCMC
    ndx0 <- loc1:loc2
    is.plotted <- !(ndx0 %% plotevery)
    is.plotted[1] <- is.plotted[nMCMC] <- T
    ndx <- ndx0[is.plotted]
    points(ndx, x[ndx], type="l",col="blue" )
  }
}

```



```

convergencePlot <- function(
  x,
  MCMCspecs,
  chainsUse=1:MCMCspecs$chains,
  burnin=MCMCspecs$nMCMC/2)
{
  # Gelman diagnostic plot with burn-in
  # all chains should be in one vector, num. &length of chains in 2nd argument

  require(coda) # multiple functions used

  chains <- MCMCspecs$chains
  nMCMC <- MCMCspecs$nMCMC
  niters <- length(x)

  #-- lower plot
  #* reformat to list of chains, delete burnin

  numuse <- length(chainsUse)
  chainlist <- vector(numuse,mode="list")

  for(i in 1:numuse) {
    ch.i <- chainsUse[i]
    loc1 <- (ch.i-1)*nMCMC + burnin + 1
    loc2 <- ch.i*nMCMC
    chainlist[[i]] <- mcmc(x[loc1:loc2])
  }

  chainlist <- mcmc.list(chainlist)
  cat("\ncoda object created\n")
  summary(chainlist)
  gelman.plot(chainlist,auto.layout=F)
}

medianByChain <- function(
  x, # chains end-to-end
  MCMCspecs, # list includes chains and num. per chain
  threshold=NULL # if value return logical vector (median > value)
)
{
  # Medians by chain, mostly for log-likelihood
  # if threshold is not null then should be scalar and
  # output vector is transformed to logical, T/F according
  # to whether chain median exceeds threshold
  # D. Farrar 7/2006

  chains <- MCMCspecs$chains; nMCMC <- MCMCspecs$nMCMC

  #-- median log-likelihood by chain
  medians <- rep(NA, chains)

  for(i in 1:chains) {
    loc1 <- (i-1)*nMCMC + 1 # first index in chain
    loc2 <- i*nMCMC # last index in chain
    medians[i] <- median(x[loc1:loc2])
  }; cat("\n") # for(i ... [over chains]

  if(!is.null(threshold)) return(medians >= threshold)
  else return(medians)
}

```

```

mbssbivasmooth <- function(data.x, data.y, data.z,
  gridoffset=5,
  gridfine=400,
  xlab="Urbanization(%)", ylab="Agriculture(%)",
  pl=T,
  imageplot=F,
  plotPoints=T,
  plotRugs=T,
  cex.lab=1,
  title=""
)
{
  # plot a bivariate smooth where x and y ranges are 0-100
  # as used in my analysis of maryland data.
  # Returns matrix of interpolated surface values (z) with x and y
  # in dimension names.
  # D. Farrar 5/2006

  require(akima)
  oldpar <- par()
  par(cex.main=1.5)
  par(col.lab="blue")
  gridside <- (-gridoffset:gridfine)*(100/gridfine)
  z <- interp(x=data.x,y=data.y,z=data.z,
    xo=gridside,yo=gridside,duplicate="mean")$z
  dimnames(z) <- list(gridside,gridside)
  if(pl) {
    if(imageplot) {
      image( gridside,gridside,z,axes=F,xlab=xlab,ylab=ylab)
    } else contour(gridside,gridside,z,axes=F,xlab=xlab,ylab=ylab,col="red",
      levels=1.6+0.2*(0:8),
      cex.lab=cex.lab,
      labcex=1,
      main=title,
      # levels=c(1.5,2,2.5,2.75,3,3.25)
    )
    if(plotPoints) points(data.x,data.y,col="blue",pch="+",cex=0.7)
    axis(side=1,at=seq(0,80,by=20))
    axis(side=2,at=seq(0,80,by=20))
    if(plotRugs) { rug(data.x,side=3,col="blue")
      rug(data.y,side=4,col="blue") }
  }
  return(z)
}

```

```

PC.eccp <- function(eccp,k,scree=T)
{
  # Preliminary.
  # principal coordinates based on eccp, k=2-4, 2 axes
  # Farrar, 2006

  GowerSimilarity <- function(S) {
    # Gower transformation of similarity matrix
    n <- nrow(S)
    I <- diag(n)
    Jbar <- matrix(1,n,n)/n
    return((I-Jbar) %*% S %*% (I-Jbar))
  }

  Gsim <- GowerSimilarity(eccp)
  spectrm <- eigen(Gsim)
  eignvecs <- spectrm$vectors
  eignvals <- spectrm$values
  dimnames(eignvecs) <- list(rownames(eccp),paste("Eig", 1:nrow(eccp)))

  if(F) {
    # par(mfrow=c(1,1))
    plot(eignvals, ylab="Eigenvalue")
    abline(h=0,lty="dotted")
  }

  #-- PC plot with circles on cluster locations

  if(T)
  {
    xylim <- c(min(eignvecs),max(eignvecs))
    plot(eignvecs[,1]/sqrt(eignvals[1]),
         eignvecs[,2]/sqrt(eignvals[2]),
         type="n",xlim=xylim,ylim=xylim,
         xlab="Dimension 1",ylab="Dimension 2")
    text(eignvecs[,2],eignvecs[,1],labels=rownames(eccp),col="blue",cex=0.7)
    dendrostuff <- eccpDendrogram(eccp,k=k,plotdendro=F)
    cluslocs <- aggregate(data.frame(eignvecs),
                          by=list(as.vector(dendrostuff$Class)),mean)[,-1]
    points(cluslocs[,2],cluslocs[,1],cex=5)
  }
  #--return
  return(eignvecs)
}

```

```

plotMultipleChains <- function(
    x,          # multiple chains in one vector
    MCMCspecs,
    xlab = "Index in Posterior Sample",
    ylab = NULL, main=NULL )
{
  # Plots multiple chains demarcated by vertical lines.
  # More general and more crude than logLikPlot, but same idea
  # (D. Farrar 4/2006)

  chains <- MCMCspecs$chains
  nMCMC <- MCMCspecs$nMCMC
  niters <- length(x)
  #-- plots
  plot(1:niters, x, type = "l", col="blue", xlab=xlab, ylab=ylab,main=main)
  # vertical lines demarcate chains on plot
  if (chains > 1) for (i in 1:(chains - 1)) abline(v = i * nMCMC, lty = 2)
}

```

```

postprocess.mcmc <- function( # with user specified function and alignment
  beta.sample, # sample of beta's (list q*K matrices).
  MCMCspecs, # mcmc specs (list(chains,nMCMC,...))
  burnin=MCMCspecs$nMCMC/2, # deleted from each chain (1*1)
  alignment=NULL, # alignment (list(ispermuted,algnmtx)).
  myfunc=function(beta) return(NULL) # vector valued function of q*K matrix
)
{
  # Generate a user-specific matrix from mcmc sample based on function myfunc()
  # * multiple chains and burnin deleted from each chain
  # * alignment specified in a list as generated by function align()
  # * output is matrix num.iterations*length(myfunc())

  niters <- length(beta.sample); # combined length of chains including burnin
  betal <- beta.sample[[1]]; # to extract q and K
  q <- nrow(betal); # num. params
  K <- ncol(betal); # num. classes
  chains <- MCMCspecs$chains; # num. chains
  n.use <- niters -chains*burnin; # sample size after deleting burnin
  nMCMC <- MCMCspecs$nMCMC; # length of each chain
  outmtx <- NULL # output matrix
  outlocij <- 0; # location in output

  #-- determine if alignment is requested / needed
  if(is.null(alignment)) do.align <- F else do.align <- any(alignment$ispermuted)

  for(i in 1:chains) {
    for(j in 1:nMCMC) { # within chain
      if(j > burnin) {
        if(!(j %% 500)) cat("\n",i,j);
        inlocij <- (i - 1)*nMCMC + j; # input location
        outlocij <- outlocij + 1; # output location
        beta.ij <- beta.sample[[inlocij]];
        if(do.align) {
          if(alignment$ispermuted[inlocij]) {
            if(K==2) {
              beta.ij <- beta.ij[,c(2,1)]
            } else {
              beta.ij <- beta.ij[,alignment$algnmtx[inlocij,]]
            }#if(K..
          }#if([permuted]...
        }#if(do.align..
        out.ij <- myfunc(beta=beta.ij);
        if(outlocij==1) outmtx <- matrix(NA,n.use,length(out.ij));
        outmtx[outlocij,] <- out.ij ;
      }#if(j>..
    }#within chain
  }#chains]
return(outmtx) } #-- end function defn. -----

```

```

rDirichlet1 <- function(alpha)
{
  ### sample of size 1 from a Dirichlet(alpha) distribution.  Divide the
  ### K independent gamma(alpha[i]) variables by their sum.

  K <- length(alpha);
  rg <- rgamma(n=K,shape=alpha); # K gamma variates w/different alpha(scale=1)
  rD <- rg[1:(K-1)]/sum(rg);

  return(c(rD,1-sum(rD)));
} #--- end defn. 'rDirichlet1' -----

```

```

replace.clusterLM <- function( # replace regression results for one cluster

  lmstuff,      # list of regression results as generated by clusterLM()
  clusindex,    # index of cluster to update (in 1,...,N) (1*1)
  Xc,           # matrix of predictors for cluster, with no col 1's (#clus*p)
  Yc,           # vector for clus (#clus*1)
  beta.fails=function(x) any(is.na(x)), # failure constraint
  locn.model=F # T then ignore Xc and compute mean

)
{
  # D. Farrar 7/2006
  # Update list of linear model results for a class indicated
  # for a class indicated in a function argument.

  N <- length(Yc)
  dgn.c <- rep(1,N)

  fails <- F;

  if(locn.model) {

    XPXinv.c <- 1/N
    SSR.c <- sum((Yc - beta.c)^2)

  } else {

    lm.c <- lm(Yc~Xc)
    beta.c<- lm.c$coefficients
    fails <- beta.fails(beta.c)

    if(!fails) {
      SSR.c <- sum(lm.c$residuals^2)
      dgn.c <- cbind(dgn.c, Xc)
      XPXinv.c <- solve(crossprod(dgn.c))
    }

  }

  if(fails) {

    cat("\nmodel rejected in replace.clusterLM\n")
    return(list(fails=T))

  } else {

    retlist <- lmstuff
    retlist$SSR[clusindex] <- SSR.c
    retlist$beta[,clusindex] <- beta.c
    retlist$XPXinv[[clusindex]] <- XPXinv.c
    retlist$dgnList[[clusindex]] <- dgn.c

    return(retlist)

  }

} #-- end fn. defn. -----

```

```

rpartition <- function(
  N, K,          # Partition N units into K clusters
  minn=1,       # subject to a minimum clusters size of minn,
  p=NULL,       # where p is an N*K matrix or N-vector of class probs.
  force=NULL    # Vector (K) of GCU indices, force jth into cluster j.
)
{
  # Randomly partition subject to minimum clusters sizes, optionally
  # based on a vector or matrix of class probabilities, or forced
  # membership of some units.
  # (D. Farrar 4/2006)

  if(is.null(p)) p <- rep(1,K)/K;
  if(!is.matrix(p)) p <- matrix(rep(p,N),N,K,byrow=T);
  ok <- FALSE;          # init. classfn. ok if min cluster sizes met

  while(!ok) {         # loop until ok

    Class <-rep(NA,N) ;      # cluster num for each unit (N*1)

    for (i in 1:N) {       # loop over CUs assigning each to cluster
      if(!is.null(force) & is.element(i,force)) {
        Class[i] <- which(force==i)
      } else Class[i]<-sample(1:K,1,prob=p[i,])
    } # for(i..

    n <- as.vector(table(Class))
    ok <-length(n)==K & (min(n) >= minn);    # ok if min cluster sizes met

  } # while(!ok)[loop until ok]

  return(list(Class=Class,n=n));
} #-- end fn. defn. -----

```



```

sortmatrix<-function(mtrx, colnum)
{
  # utility function orders rows in a matrix (arg.1) so that values in a
  # according to values in designated col (arg. 2) (non-decreasing).
  # Transfer row names if present.
  # Farrar 8/2004

  r <- rank( mtrx[,colnum], ties.method="first" );
  nr <- nrow(mtrx);
  nc <- ncol(mtrx);
  mtrx2 <- matrix(NA,nr,nc);
  if(!is.null(colnames(mtrx))) colnames(mtrx2)<-colnames(mtrx);
  if(!is.null(rownames(mtrx)))
  {
    rnams <- rownames(mtrx);
    rownames(mtrx2)<-rep("",nr);
    for(i in 1:nr) rownames(mtrx2)[r[i]] <- rnams[i] ;
  }
  for(i in 1:nr) mtrx2[r[i],] <- mtrx[i,];
  mtrx2
} #----- end definition 'sortmatrix' -----

```

```

transectplot <- function(
  zmtx,                # as for image plot (z for x-y grid)
  percset,            # agriculture values
  transpose,         # T for urban F for agri
  xlab="", ylab="BIBI",
  linelab="Agr.",    # string part of curve label
  labelcurves=F,     # add labels to curves (T/F)
  labelmanual=F,     # manual placement of line labels with locator()
  title="",
  add=F,
  linecolor="red",
  xlim=c(-10,80), ylim=c(1,4) # mbss defaults
)
{
  # plot rows from matrix defining 3D surface
  # presently specific to mbss, defaults urbanization by agriculture

  curvelabs <- c("a","b","c","d","e","f")

  if(transpose) zmtx.use <- t(zmtx) else zmtx.use <- zmtx
  gridside <- as.numeric(rownames(zmtx.use))
  plotprofiles <- zmtx.use[is.element(gridside,percset),]
  print(rownames(plotprofiles),quote=F)
  if(!add)
  {
    plot(gridside,plotprofiles[1,], type="n",
         xlab=xlab, ylab="BIBI", xlim=xlim, ylim=ylim, cex.lab=1, axes=F,
         main=title)
    abline(h=2.5,lty="dotted")
    axis(side=1,at=seq(0,80,by=20))
    axis(2)
  }
  for(i in 1:length(percset)) {
    pfl.i <- plotprofiles[i,]
    cat("\n",i,rownames(plotprofiles)[i])
    if( labelcurves ) { # label lines (T/F)
      points(gridside,pfl.i,type="l",col=linecolor)
      if( labelmanual ) {
        labelWithPointer( gridside,pfl.i,
                          paste(linelab,percset[i],"%",sep=""),xlim, ylim )
      } else text(-4,na.omit(pfl.i)[1],percset[i],col="blue")
    } else {
      labvals <- c(range(gridside),10*(0:10))
      mtx <- cbind(gridside,pfl.i)[is.element(gridside,labvals),]
      points(mtx[,1],mtx[,2],type="b",pch=curvelabs[i])
    }
  }
  }#for
  if(labelcurves & !labelmanual) {
    cat("\n[waiting] click label locn.\n")
    text(-5,locator(1)[2],paste(linelab,"(%)",sep=""))
  }
}#end fn defn

```

```

zofgamma <- function(classification,u=sort(unique(classification)),
                    unitlabels=NULL)
{
  # Convert representation of a partition from class indices to
  # binary indicators, as for function 'unmap' in package 'mclust'
  # (but no 'noise' argument). First arg. is vector of class
  # indices per observation. Optional second arg. gives the set
  # of component indices, sorted.

  if(!(max(classification)==length(u))) stop("\n(zofgamma)")
  N <- length(classification)
  K <- length(u)
  z <- matrix(0, N, K, dimnames=list(NULL,paste("Cl.",u,sep="")))
  for (j in 1:K) z[classification==u[j],j] <- 1
  rownames(z) <- unitlabels
  return(z)
}

```

## Appendix II. R Functions Used in Clustering of Temporal Profiles.

The functions displayed were used for the clustering of temporal profiles in Chapter 5.

The functions are not beta tested.

```
dendpl <- function(
  distmtx,          # symmetric matrix of distances
  method,          # 'average', 'complete', 'ward', etc.
  leaflabels=NULL, # leaf labels
  distlabel        # axis label depends on distance
)
{
  # Generation and customized plotting of dendrogram using
  # R function hclust.
  # D. Farrar 5/2006

  D <- as.dist(distmtx)          # base R function returns distance matrix
  dendro <- hclust(D, method=method) # cluster object
  dendro.relabeled <- dendro
  if(!is.null(leaflabels)) dendro.relabeled$labels <- leaflabels
  plot(dendro.relabeled,ylab=distlabel,sub="",xlab="",main="",cex=0.75)

  return(list(dendro=dendro,
             dendro.relabeled=dendro.relabeled))
} # -- end fn defn -----

distance.Z2 <- function(Z)
{
  # computes matrix of square differences from Z matrix
  # Input is a vector of Z statistics with names
  # D. Farrar 5/2006

  if(sum(is.na(Z))) stop("distance.Z2 - missing input not permitted.")
  statiset <- names(Z)
  distmtx <- matrix(0,nstation,nstation,dimnames=list(statiset,statiset))
  nstation <- length(Z)
  for(i in 1:(nstation-1))
    for(j in (i+1):nstation)
      distmtx[i,j] <- distmtx[j,i] <- (Z[i] - Z[j])^2

  return(distmtx)
} #-- end fn dfn. -----
```

```

Ktau <- function(X,minN=2) {

  # Kendall's tau and related statistics for multiple strata, e.g., stations.
  # Arguments:
  # 1) data matrix where e.g. rows correspond to stations and columns to years.
  #    Missing values are permitted.
  # 2) minimum number (non-missing) for each station
  # D. Farrar 5/2006

  numLocs <- dim(X)[1]
  Z <- S <- tau <- rep(NA,numLocs) # initialize
  names(Z) <- rownames(X)
  for(k in 1:numLocs)           # loop over stations
  {
    kData <-X[k,]                # data for station k in kth row
    miss01<-is.na(kData)        # 1 or 0 according as missing or not
    Nk <-sum(!miss01)            # years w/ data not missing for locn k
    if(Nk >= minN)              # use locns w/at least the min num yrs
    {
      kData <- kData[!miss01]    # non-missing values for location
      S.k <- pairs.k <- 0        # S, number of non-tied pairs
      for(i in 1:(Nk-1))
      {
        for(j in (i+1):Nk)
        {
          x.i <-kData[i]
          x.j <-kData[j]
          if (x.i != x.j)        # if [not tied]
            pairs.k <- pairs.k + 1
            S.k <- S.k + ((x.i <= x.j) - (x.j <= x.i))
        } # for(j ... [over rows]
      } # for(i ... [over rows]
      varSk <- Nk*(Nk-1)*(2*Nk+5)/18 # variances without tie adj
      forAdj<- as.vector(table(kData)) # counts per tied group
      is.gt1<- (forAdj > 1)          # flag tied groups size gt 1
      if(any(is.gt1)) {             # then do tie adjustment
        forAdj<- forAdj[is.gt1]
        varSk <- varSk - (1/18)*sum(forAdj*(forAdj-1)*(2*forAdj+5))
      } # if(any( ... [tie adj]
      tau[k] <- S.k / pairs.k
      Z[k] <- (S.k - (S.k>0) + (S.k<0))/sqrt(varSk)
    } # if(Nk>=minN) [enough data for stats ]
  } # for k in 1...[over stations]

  return(list(Z=Z,tau=tau))
} #-- end fn. defn. -----

```

```

plotEnvelope <- function(
  x,                # (N*T) matrix where each row is a profile
  xforplot,        # year series
  forEnvelope=NULL, # (N*T) logical - which rows to use for envelope
  asBackground=NULL, # (N*T) logical - which plot as lines in background
  asForeground=NULL, # (N*T) logical - which plot as lines atop envelope
  plotMedian=F,    # Logical - plot only medians for envelope set?
  ylim=NULL, xlab="year", ylab=NULL, main="" # graph params
)
{
  # superimpose plots of multiple series, divided into 3 subsets,
  # any of which may be empty.  Optionally include median
  # of envelope set.
  # remark: I like to log10 the ordinate, but I do that external to
  # the function, and pass the logs to the function. (DF)(
  # D. Farrar 6/2006.

  numprofiles <- nrow(x)

  #-- line type for foreground lines

  if(plotMedian) lty <- "dashed" else lty <- "solid"

  #-- plot axes only

  plot(xforplot,x[1,],type="n", ylim=ylim, xlab=xlab,ylab=ylab, main=main)

  #-- dotted lines for background profiles

  if(!is.null(asBackground)) for(i in 1:numprofiles) if(asBackground[i])
    points(
      xforplot,x[i,],type="l",lty="dotted",col="maroon4"
    )

  #-- envelope for specified set

  if(!is.null(forEnvelope)) {
    forEnvM <- x[forEnvelope,]
    envL <- apply(forEnvM,2,function(x) min(x,na.rm=T))
    envU <- apply(forEnvM,2,function(x) max(x,na.rm=T))
    pgon <- rbind( cbind( xforplot, envL),
                  cbind( rev(xforplot), rev(envU))
                )
    polygon(pgon,col="azure",border=plotMedian)
    # points(xforplot,envL,type="l")
    # points(xforplot,envU,type="l")
  }

  #-- foreground profiles

  if(!is.null(asForeground)) for(i in 1:numprofiles) if(asForeground[i])
    points(
      xforplot,x[i,],type="l",lty=lty
    )

  #-- line for means

  if(plotMedian)
    points(xforplot,apply(forEnvM,2,function(x) median(x,na.rm=T)),type="l")
}

```

```

simi.RTepsi <- function(
  dmtx,          # data matrix stations * years with row names
  minyrs=2      # required minimum years data with values in both profiles
)
{
  # Compute a matrix of RTepsi subject to minimum count of years in common
  # with values not missing for each pair of stations. In case of missing
  # values in the matrix, delete stations until there are none missing.
  # Calls XCorrFunc for each pair of stations.
  # D. Farrar 5/2006

  dmtx <- as.matrix(dmtx)
  statiset <- rownames(dmtx)
  nstation <- nrow(dmtx)
  XCorrM <- matrix(NA,nstation,nstation,dimnames=list(statiset,statiset))
  diag(XCorrM) <- 1
  for(i in 1:(nstation-1)) {
    y.i <- as.vector(dmtx[i,])
    for(j in (i+1):nstation) {
      y.j <- as.vector(dmtx[j,])
      canUse <- !(is.na(y.i)|is.na(y.j))
      if(sum(canUse) >= minyrs) {
        both <- cbind(y.i,y.j)[canUse,]
        c.ij <- XCorrFunc(both[,1],both[,2])
        XCorrM[i,j] <- XCorrM[j,i] <- c.ij
      }#if
    }# for(j ...)
  }# for(i ...)

  if(any(is.na(XCorrM))) {
    cat("\nDeleting stations from RTepsi matrix\n")
    while(any(is.na(XCorrM))>0) {
      n <- ncol(XCorrM)
      if(n==1) {
        stop("!! too many stations excluded")
      } else {
        numnotNA <- apply(XCorrM,1,function(x) sum(!is.na(x)))
        delstatn <- which.min(numnotNA)
        cat("\ndeleted:",colnames(XCorrM)[delstatn])
        XCorrM <- XCorrM[-delstatn,-delstatn]
      }#if
    }#while
  }#if

  return(XCorrM)
} #-- end fn dfn. -----

```

```

vbh <- function(Z,loquacious=T) {

  # van Belle & Hughes test for heterogeneity of trend.
  # Input vector of Z's (some missing ok). Returns p-value.
  # If second arg is TRUE then print some output.
  # D. Farrar 5/2006

  has.Z <- !is.na(Z) # id which not missing
  m <- sum(has.Z)
  chiSqHom <- pHomo <- pTotal <- NA
  if(m >= 1) {
    Z <- Z[has.Z]
    chiSqTot <- sum(Z^2) # total chi-square
    chiSqTrd <- m*mean(Z)^2 # trend chi-square
    pTotal<- pchisq(chiSqTot, m, lower.tail=F)
    if(m >= 2) {
      chiSqHom <- chiSqTot - chiSqTrd # homogeneity chi-square
      pHomo <- pchisq(chiSqHom, m-1, lower.tail=F)
    }
  }

  if(loquacious) cat(
    "\nchi-square\nTotal\t",chiSqTot,"\nTrend\t",chiSqTrd,"\nHomog\t",
    chiSqHom,"\np=\t",pHomo )

  return(list(pHomo=pHomo, chiSqHom=chiSqHom, pTotal=pTotal))
} #-- end fn. defn. -----

```



```

XCorrFunc <- function(v1,v2,minNotNA=2) {

  # Kendall-type cross-correlation between two series
  # 3rd arg is minimum num not missing in both vectors
  # input vectors must be equal in length.
  # See also : simi.RTepsi uses for computing distance matrix.
  # D. Farrar 5/2006

  nonmissing <- !(is.na(v1)|is.na(v2))
  v1 <- v1[nonmissing]
  v2 <- v2[nonmissing]
  n <- length(v1)

  if(n < minNotNA) {
    retval <- NA
    if(minNotNA==2)
      cat("\n(XCorrFunc): !!comparing series of length < 2\n")
  } else {
    npairs<-n*(n-1)/2      # pairs with neither missing
    concord <- 0          # count of concordant pairs
    for(i in 1:(n-1)) {
      for(j in (i+1):n)
        concord <- concord +
          ( (v1[j]-v1[i])*(v2[j]-v2[i]) > 0 ) +
            ( (v1[j]==v1[i])&(v2[j]==v2[i]) ) )
    } # for(i ...
    retval <- concord / npairs
  }

  return(retval)          # fraction of pairs concordant
} #---[ end fn. defn. ]-----

```