MONOTHETIC CLUSTER ANALYSIS WITH EXTENSIONS TO CIRCULAR AND FUNCTIONAL DATA

by

Tan Vinh Tran

A dissertation submitted in partial fulfillment
of the requirements for the degree

of

Doctor of Philosophy

in

Statistics

MONTANA STATE UNIVERSITY
Bozeman, Montana

May 2019

DEDICATION

I dedicate this dissertation to my parents, my wife, Thư, and my son, Thiện.

Xin dành tặng công trình này cho Ba Mẹ, cho vợ, Thư, và con trai, Thiện.

ACKNOWLEDGEMENTS

I would like to express my gratitude to my advisor, Dr. Mark Greenwood, for his invaluable guidance and patience throughout the course of my Ph.D. studies. His insightful vision of the details in this dissertation and his intensely observant assessment of my writing facilitated a complete product that I can be proud of.

I would like to thank Dr. John Borkowski for his tireless encouragement even before I came to this university. Without you, I would not be here. I would also like to express my gratitude to my former and current committee members: Dr. Megan Higgs, Dr. Laura Hildreth, Dr. Nicole Carnegie, and Dr. Jim Robison-Cox. I deeply appreciate your support and invaluable comments on this dissertation that enable me to consolidate my future publications.

I am also grateful for the support from the Vietnam Education Foundation, the Department of Mathematical Sciences, and the Statistical Consulting and Research Services. Without their fellowship and assistantship, my education at Montana State University would not be possible.

Finally, I wish to thank my dad, mom, brother, wife, and son for their love and constant support to make this journey a reality.

TABLE OF CONTENTS

TABLE OF CONTENTS – CONTINUED

LIST OF TABLES

LIST OF FIGURES

LIST OF FIGURES – CONTINUED

LIST OF FIGURES – CONTINUED

LIST OF FIGURES – CONTINUED

LIST OF FIGURES – CONTINUED

LIST OF ALGORITHMS

NOMENCLATURE

Q	number of response variables
n	Sample size
y_{iq}	the i^{th} observation on variable q
$\mathbf{y_i}$	the i^{th} observation, a vector of $(y_{i1}, y_{i2}, \ldots, y_{iQ})$
$C_1, C_2, \ldots, C_K$	the specific clusters in a K-cluster solution
Ω	a data set
C_k	a cluster
C_{kL} and C_{kR}	children clusters of a cluster C_k
$s(C_k)$	a binary split s to the cluster C_k
$\Delta I(s, C_k)$	The change in inertia when applying the split s to a cluster C_k
$d(\mathbf{y_i}, \mathbf{y_j})$	the distance (dissimilarity) between two observations $\mathbf{y_i}$ and $\mathbf{y_j}$
M	number of folds in M-fold cross validation

ABSTRACT

Monothetic clustering is a divisive clustering method that uses a hierarchical, recursive partitioning of multivariate responses based on binary decision rules that are built from individual response variables. This clustering technique is helpful for applications where the rules of groupings of observations as well as predicting new subjects into clusters are both important. Based on the ideas of classification and regression trees, a monothetic clustering algorithm was implemented in R to allow further explorations and modifications.

One of the common problems in performing clustering is deciding whether a cluster structure is present and, if it is, how many clusters are "enough". Some well-established techniques are reviewed as well as new methods based on cross-validation and permutation-based hypothesis tests at each split are suggested.

Monothetic clustering is of interest to be applied in a variety of situations. This can include data sets with circular variables, where the variables' natures are not linear. A method for monothetic clustering and visualizations of clusters with circular variables was developed that could also be used in other classification and regression tree situations. Clustering is also interesting for data sets where the responses can be transformed into functional data, which has unique properties that need exploring. Partitioning Using Local Subregions (PULS), a clustering technique inspired by monothetic clustering to overcome some of its disadvantages in clustering functional data, is discussed. In this algorithm, clusters are formed based on aggregating the information from several variables or time intervals. In both monothetic clustering and PULS, it is possible to limit the set of feasible splitting variables to be able to create clusters for new observations without observing all variables or times to assign new observations to the clusters.

R packages for these methods have been developed for others to use and test and support the proposed research, and a detailed vignette is provided for utilizing all the functions developed here.

CHAPTER ONE

INTRODUCTION

Abstract: Cluster analysis, or clustering, is a set of techniques for dividing data into groups, or clusters, that both maintain the internal cohesion within clusters and the external isolation between clusters. It is widely used in statistics, data mining, and machine learning with applications in data exploration, marketing, medical diagnostics, computational biology, and many other areas. A specific type of clustering, monothetic clustering, that uses a hierarchical, recursive partitioning of multivariate responses based on binary decision rules that are built from individual response variables, is the main theme of this dissertation, where we explore its performance and applications to special data such as circular variables and functional data. This chapter reviews the literature that paves a way for methods mentioned in other chapters, including clustering, circular data analysis, functional data analysis, and available statistical packages for those techniques in R.

1.1 Cluster Analysis

Cluster analysis, or clustering, is an unsupervised learning technique that attempts to group subjects based on (multivariate) observations into clusters so that the dissimilarity within clusters is smallest while the between cluster dissimilarities are largest. The purpose of clustering is to identify hidden patterns where little or no information about those patterns in the data are known. Clustering is widely used in statistics, data mining, and machine learning with applications in data exploration, marketing, medical diagnostics, computational biology, and many others.

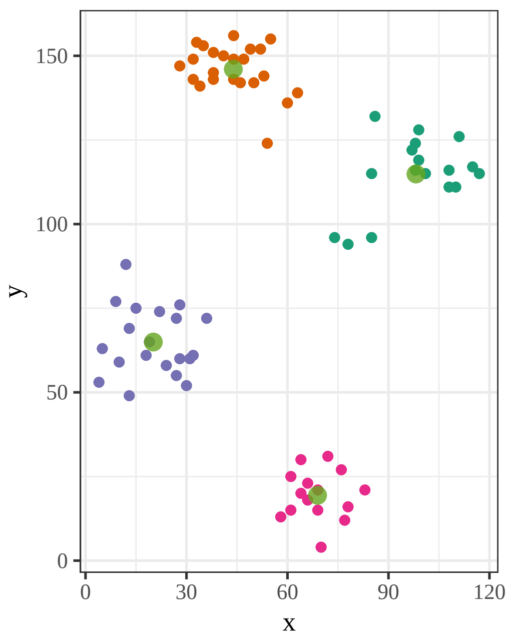
Let y_{iq} be the i^{th} observation on variable q with $q = 1, \ldots, Q$, where Q is the number of response variables and $i = 1, \ldots, n$ where n is the number of observations. We seek to divide the n objects into partitions $C_1, \ldots, C_k, \ldots, C_K$, where K is the number of clusters and C_k is the k-th cluster. As a simple example to illustrate cluster analysis, we consider an artificial data set introduced by Ruspini (1970) with 75 observations on two variables, x and y. The scatterplot of the data set is shown in Figure 1.1 and a potential four cluster result of k-means and k-medoids (mentioned below) are also plotted. In this simple example with only two variables, a cluster includes data points that are "similar" to each other and "dissimilar" to data points in other clusters. In higher dimensional data sets, the dissimilarities between data points are more difficult to visualize, so a dissimilarity measure must be defined, with choices such as Euclidean, Gower's, Manhattan, and Mahalanobis often considered. Commonly used clustering algorithms include k-means, k-medoids, and Ward's agglomerative hierarchical method, which are discussed below.

The k-means algorithm for a K group cluster analysis (MacQueen, 1967) starts by initializing the K data points (called K cluster centers, or centroids). The *assignment step* starts by assigning other data points to one of the K centers which has the smallest squared Euclidean distance, in which the distance between two data points $\mathbf{y_i}$ and $\mathbf{y_j}$ is defined as

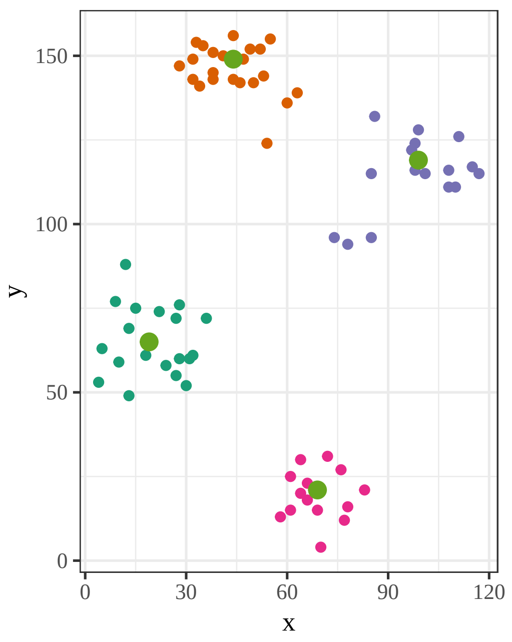
$$d(\mathbf{y_i}, \mathbf{y_j}) = d^2_{euc}(\mathbf{y_i}, \mathbf{y_j}),$$

where $d_{euc}(\mathbf{y_i}, \mathbf{y_j})$ is the Euclidean distance between two data points $\mathbf{y_i}$ and $\mathbf{y_j}$,

$$d_{euc}(\mathbf{y_i}, \mathbf{y_j}) = \sqrt{\sum_q (y_{iq} - y_{jq})^2}. \tag{1.1}$$



(a) Four cluster result by k-means.



(b) Four cluster result by k-medoids.

Figure 1.1: Scatterplot of the *ruspini* data set with two variables x and y. The data points are colored by the cluster membership. Centroids and medoids are plotted double in size and in different colors.

After this step, K groups (clusters) of data points have been formed. The centroid $\overline{y}_{C_k}$, the center point of cluster C_k, is re-calculated as the mean of all data points in that cluster, which is

$$\overline{y}_{C_k} = \frac{\sum_{i \in C_k} \mathbf{y_i}}{n_k},$$

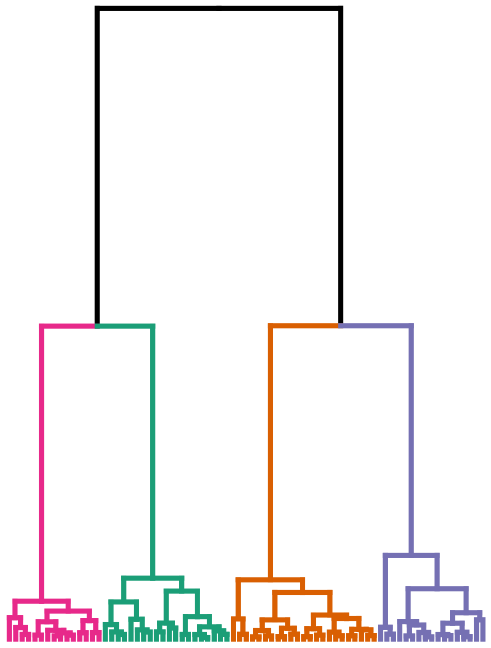
where n_k is the number of the data points in cluster C_k. This is called the *update step*. It then repeats the *assignment step* by re-assigning data points to their closest centroids. The *update* and *assignment steps* are repeated until there is no more change in the assignment step. By doing this, k-means clustering tries to minimize the within-cluster sum of squares and, at the same time, maximize the between-cluster sum of squares. k-means clustering has a straightforward algorithm, but it has some disadvantages. The problem of minimizing the total squared distance within clusters is a non-deterministic polynomial-time (*NP*) hard. Lloyd's algorithm (Lloyd, 1982), which is commonly mentioned in textbooks (e.g., Hastie et al., 2016; James et al., 2013), uniformly randomly selects initial centroids from the clustering data points. However, the clustering result can be locally optimal and is not guaranteed to be the global optimum. Moreover, the clustering result can be different for each run, depending on the initial K data points. This algorithm has been examined extensively since it was introduced. Some researchers suggested modifications on the algorithms such as introducing various algorithms to pick the initial points (Celebi et al., 2013) or methods to avoid local optima (for example, a method suggested by Hartigan and Wong, 1979). For more details on some of the research on k-means, see Everitt et al. (2011, p 125).

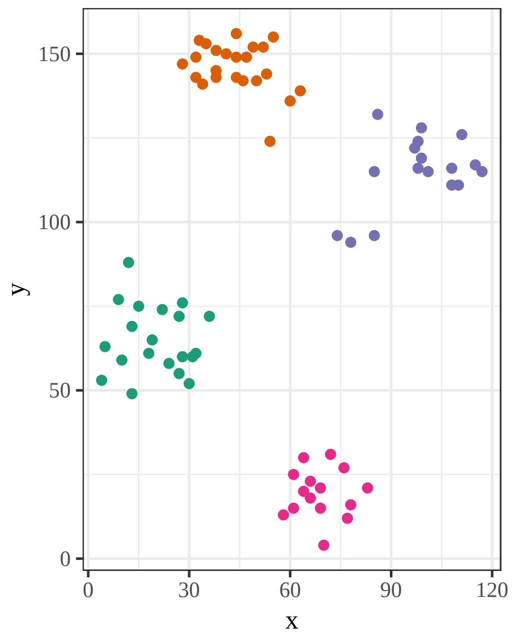
Kaufman and Rousseeuw (1990) suggested using actual observations as the representatives of the clusters (which they called *medoids*) instead of centroids, the means of all data points in the clusters, as in k-means. The corresponding k-medoids clustering technique then assigns other observations to their nearest medoids to form the clusters. *Partitioning around medoids* (PAM) is an algorithm suggested for performing k-medoids. PAM starts with the *build step* by choosing the initial K representative observations that have the minimum average (or sum) distance to other observations. Subsequently, other observations are considered to replace the representative ones in the *swap step* until the average (or sum) distance to other observations cannot be any smaller. Although k-medoids shares many similarities to k-means, the use of actual observations as the representative data points of the clusters provides a different optimization target, that is, to minimize the average (or sum) distance (or dissimilarity) of observations to their closest medoids. This makes k-medoids more flexible than k-means in terms of allowing any dissimilarity measure besides (squared) Euclidean distance. Moreover, k-medoids is more robust with respect to outliers (Kaufman and Rousseeuw, 1990).

Another standard clustering method is hierarchical clustering. This approach is different from k-means and k-medoids in that the number of clusters do not have to be decided *a priori*. The agglomerative (or bottom-up) hierarchical method starts by considering each individual observation as one cluster. The two clusters (observations) that are most similar to each other will then be fused with each other to form a larger cluster. The algorithm continues this step until there is only one cluster containing all observations. The clustering process can be represented in a tree, called a *dendrogram*, and the researchers can decide on the number of resulting clusters using the dendrogram. There are two issues that need to be addressed in the hierarchical methods. First, hierarchical clustering is based exclusively on the (dis)similarity between observations, so it is important to choose a suitable distance metric. The most common distance metric used is Euclidean distance. Second, the (dis)similarity measure between clusters, which is called *linkage*, needs to be defined for the algorithm to decide which two clusters to be fused in the next step. Linkages include *complete* (the greatest distance between all pairs of observations in two clusters), *single* (the smallest distance between all pairs of observations in two clusters), *average* (the average distance between all pairs of observations in two clusters), *centroid* (the distance between the centroids of two clusters), and Ward's method.

Ward (1963) introduced the criterion for fusing two clusters based on minimizing the change in the total within-cluster sum of squared distance when fusing the clusters, defined as $\sum_k I(C_k)$, where

$$I(C_k) = \sum_{i \in C_k} d^2_{euc}(\mathbf{y_i}, \overline{y}_{C_k}). \tag{1.2}$$


(a) Dendrogram.



(b) The four cluster solution.

Figure 1.2: Result of Ward's agglomerative hierarchical method on *ruspini* data.

Therefore, the target of Ward's method is similar to k-means, which is to minimize the total sum of squared distance of the clusters. According to Murtagh and Legendre (2014), there are two realizations of Ward's method target function, one that uses the sum of squared distance between observations (*Ward1*) and another one that uses the square root sum of squared distance between observations (*Ward2*). *Ward2* is the one attributed to Ward (1963). It maintains the original target function (Eq. 1.2) correctly, and is suggested as the better version of the algorithm. Murtagh and Legendre (2014) also showed that *Ward1* can produce the same results as *Ward2* if the input distance of *Ward1* is the squared input distance of *Ward2*. For example, *Ward1* using squared Euclidean distance to calculate distance matrix will result in the same clustering results as *Ward2* using Euclidean distance. The *Ward2* algorithm applied to *ruspini* data set gives a dendrogram and the four cluster result displayed in Figure 1.2.

1.2 Monothetic Clustering and Cluster Visualization

The previous three clustering methods belong to a group of methods called *polythetic clustering* (MacNaughton-Smith et al., 1964), that use combined information of variables to partition data. Clusters created are similar “on average” but may share no common characteristics. In contrast, monothetic cluster analysis (Sneath and Sokal, 1973) is a clustering algorithm that produces clusters with shared characteristics, such as the same category of a categorical response or in the same interval of a quantitative variable, using a hierarchical, recursive partitioning of multivariate responses based on binary decision rules that are built from individual response variables. Monothetic clustering is the main interest of this dissertation and is examined in different perspectives throughout the chapters.

Inspired by classification and regression trees (Breiman et al., 1984), Chavent (1998) suggested a monothetic clustering algorithm that searches for splits from each response variable that provides the best split of the multivariate responses in terms of a global criterion called *inertia*, the total variability around the cluster centroid. In the case of Euclidean distance (Equation 1.1), the inertia $I(C_k)$ for cluster C_k would be Equation 1.2.

It has been shown that the squared Euclidean distances for all observations within a cluster and variation around the means within a cluster are directly related (James et al., 2013), so the within cluster inertia can be equivalently calculated from the dissimilarity matrix as

$$I(C_k) = \frac{1}{n_k} \sum_{(i,j)\in C_k, i>j} d^2_{euc}(\mathbf{y_i}, \mathbf{y_j}), \tag{1.3}$$

where n_k is the cardinality (size) of C_k. This result is used to justify the application of many distance-based methods to non-Euclidean dissimilarities. For example, Anderson (2001) applied Equation 1.3 to Bray-Curtis dissimilarities (Bray and Curtis, 1957), a non-Euclidean semi-metric measure that does not satisfy the triangular inequality. Hence the monothetic clustering algorithm can also be directly applied to non-Euclidean distances and other dissimilarities, including data that contain circular variables (see Chapter 3), mixed data with categorical and quantitative variables, and even data with missing observations.

A binary split $s(C_k)$ on a cluster C_k divides its observations into two smaller clusters, C_{kL} and C_{kR}. The inertia decrease before and after the partition is defined as

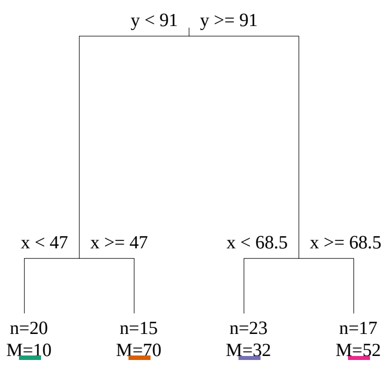
$$\Delta(s, C_k) = I(C_k) - I(C_{kL}) - I(C_{kR}), \tag{1.4}$$

and the best split $s^*(C_k)$ is the split that maximizes this decrease in inertia,

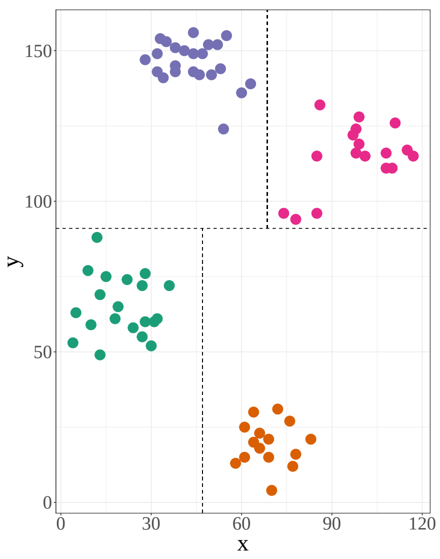
$$s^*(C_k) = \arg\max_s \Delta(s, C_k). \tag{1.5}$$

The same algorithm is then recursively applied to each sub-partition, recording splitting rules on its way. The result of the algorithm is a set of hierarchical binary rules for assigning cluster membership. Therefore, the resulting hierarchy can be read and displayed as a decision tree.

In the example of clustering the *ruspini* data to a four cluster solution, the splitting rule tree and the application of the rules to the data set can be seen in Figure



(a) Splitting rules tree.



(b) Cluster result with splitting rules.

Figure 1.3: Monothetic clustering on the *ruspini* data with four cluster solution.

1.3. The figure clearly shows that the clusters belong to four distinct quadrants. Therefore, the characteristics of each cluster can be interpreted based on the values of y (whether the observations have y larger or smaller than 91), then the values of x (if observations have $y < 47$ then the cut point is $x = 47$, and if observations have $y > 47$ then the cut point is $x = 68.5$).

Chavent et al. (2007) compared the monothetic clustering to k-means and Ward's method by simulation and application to six real-life data sets (three data sets with numerical variables and three data sets with categorical variables with known true cluster memberships) from the UCI Machine Learning repository (Dua and Graff, 2019). They found that monothetic clustering performed better than k-means and Ward's methods in terms of correct classifications of objects when the number of clusters is small. Also, for numerical data sets, the monothetic clustering had a better performance when the number of observations were larger.

In multivariate data analysis, because scatterplots are limited at 2 or 3

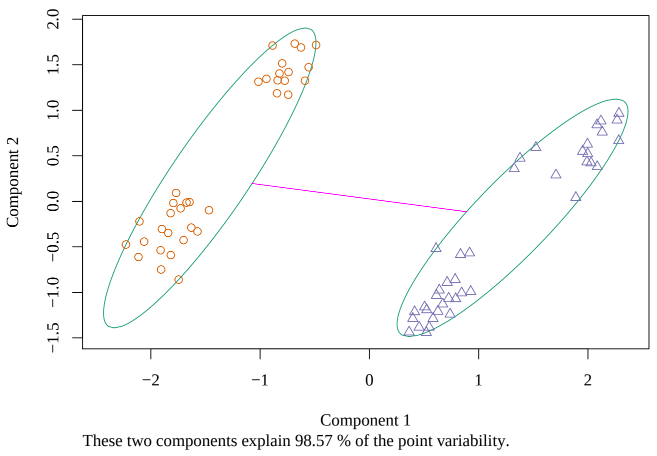
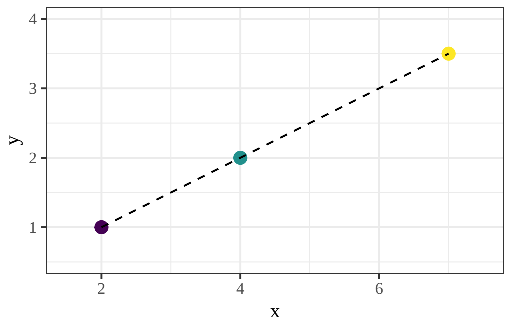


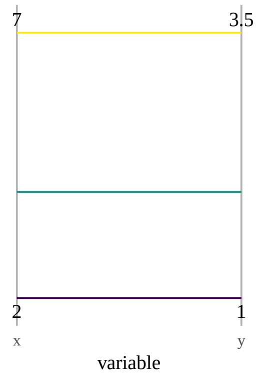
Figure 1.4: CLUSPLOT of monothetic clustering of the *ruspini* data with two clusters. The axes are the first and second principal components from the PCA of the data. We can see bimodal patterns in both clusters, indicating that the two cluster result is not sufficient for these data.

dimensions, it is not possible to visualize the results of a cluster analysis effectively when the number of variables is larger than 3. There are statistical methods to attempt to combine the variables to reduce the dimensions such as *Principal component analysis* (PCA) and *Multi-dimensional scaling* (MDS), that can be used to visualize clusters in a 2-dimensional plot such as CLUSPLOT by Kaufman and Rousseeuw (1990). The `clusplot` function in the **clusplot** package (Pison et al., 1999) can create a CLUSPLOT in R (R Core Team, 2019), with an example of plotting the *ruspini* cluster results in Figure 1.4. Although plots like CLUSPLOT can help visually check if more clusters are needed or if there is any misclassification in the cluster result, because of the dimension reduction through linear combinations of variables done by the methods, the meaning of the original variables is lost, making it more difficult to interpret the results.

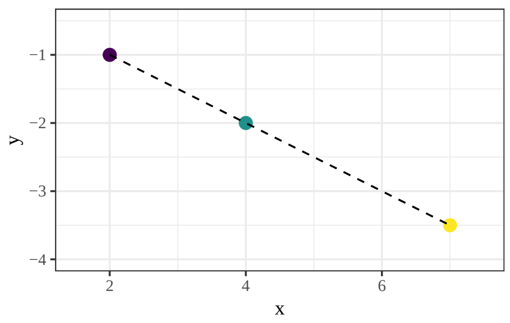
The Parallel coordinates plot (PCP, Inselberg and Dimsdale, 1987) is a means



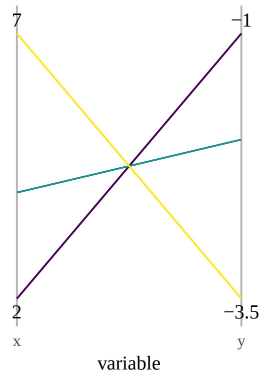
(a) Perfect positive linear relationship



(b) Parallel lines in PCP



(c) Perfect negative linear relationship



(d) crossing lines in PCP

Figure 1.5: The points in Euclidean coordinates (left) and their representations in Parallel coordinates plots (right). The variables in PCPs are scaled to have the same length with the minimum and maximum values shown.

to display multi-dimensional data where Q variables are represented by Q lines, placed equidistant and perpendicular to the traditional x-axis (hence the name parallel coordinates). A multivariate data point $\mathbf{y_i}$ $(1 \times Q)$ that has the coordinates $(y_{i1}, y_{i2}, \ldots, y_{iQ})$, a point in a Euclidean Q-dimensional space, is a polygonal line whose q-th vertex is at the y_{iq} value on the q-axis for $q = 1, 2, \ldots, Q$. Because points in Euclidean space can be displayed in a PCP as line segments (see Figure 1.5 for a visual explanation), the linear dependencies between variables can be detected. Mostly parallel lines in PCP are evidence for a positive linear relationship between

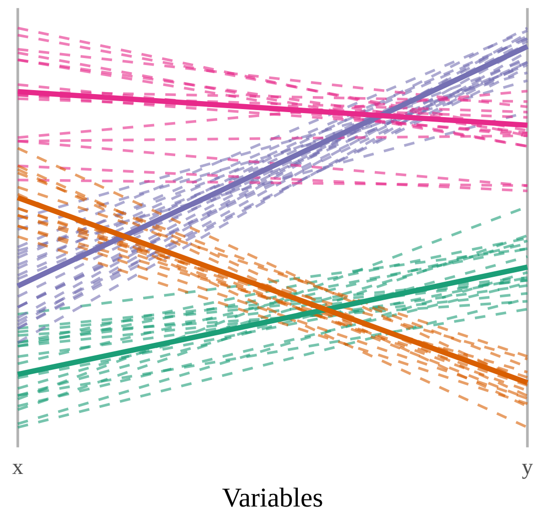
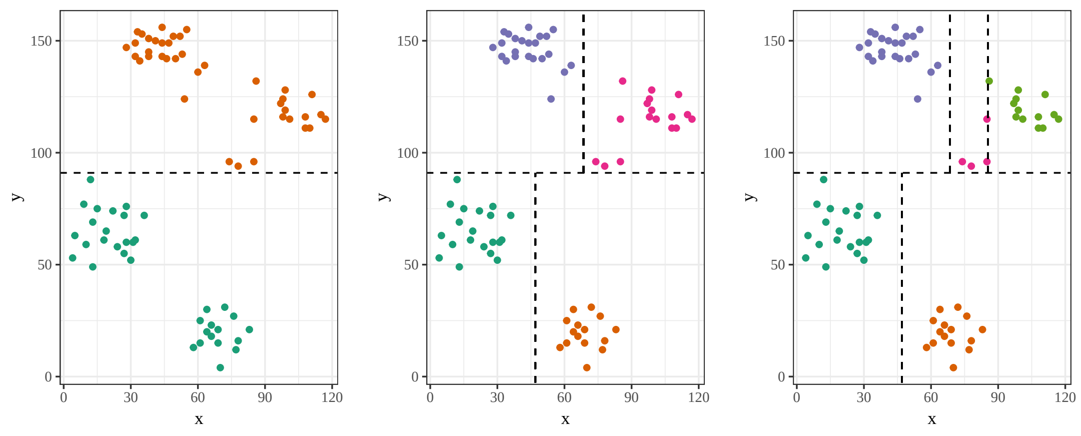


Figure 1.6: A parallel coordinates plot for the *ruspini* data. The cluster result of monothetic clustering with 4 clusters is displayed in colors. The thicker lines are the medoids of the clusters.

neighboring variables, mostly intersecting or crossing lines are evidence for a negative linear relationship, and other patterns may be discerned by tracking combinations of results across the Q variables. Another useful characteristic of a PCP as a visualization tool for cluster analysis is the ability to display and explore the identified sub-groups and highlight cluster centroids or medoids. Finally, the convergence of lines to discrete values of a (discrete or categorical) variable is visual evidence of sub-groups in the data set with common characteristics (Härdle and Simar, 2015, Chapter 1). For the *ruspini* example, the PCP of the four cluster result of monothetic clustering is shown in Figure 1.6. The values of x and y are scaled to have the same value range (same minima and maxima). The characteristics of the clusters regarding the relationship between x and y can be visualized. When a cluster result of a real data set is plotted in PCP with an appropriate choice of variable order on the x-axis and rotation (for circular variables), the interpretations can be more interesting, as in Section 3.4.2 in Chapter 3.



(a) Two clusters. (b) Four clusters. (c) Five clusters.

Figure 1.7: Different possible cluster solutions of monothetic clustering for *ruspini* data.

1.3 Choosing the Number of Clusters

One of the main benefits of cluster interpretation is to describe the shared characteristics of the members of each cluster. The choice of the number of clusters, K, greatly impacts the group memberships and, hence, the interpretation of the results. If K is too small, it puts "unlike" subjects together. On the other hand, if K is too large, it would split observations that should be together. Picking the "reasonable" K is critical for any cluster analysis. For example, possible cluster solutions of *ruspini* with different numbers of clusters using monothetic clustering can be seen in Figure 1.7. Visually, the two and four cluster solutions seem to be reasonable, but if five clusters are chosen, another cut has to be done even though the clusters are not visually separated, so it would not be recommended.

Many metrics for choosing the number of clusters have been mentioned in the clustering literature, such as a paper on the comparison of metrics by Milligan and Cooper (1985), and implemented in software such as the package `NbClust` (Charrad et al., 2014) in R. However, there is no universally "good" metric for all clustering problems or algorithms even though some are more commonly used than others. In this dissertation, we chose two classic metrics that have good performance and are suitable for monothetic clustering to compare to our new approaches.

1.3.1 Average Silhouette Width

Rousseeuw (1987) introduced the average silhouette width to quantify the structure of a cluster result. The silhouette width is a measure of how "comfortable" an observation is in the cluster it resides in. Let $a(i)$ be the average dissimilarity between observation i and other observations in the same cluster, $d(i)_k$ be the average dissimilarity between i and other observations in cluster k that is not its own cluster, and $b(i) = \min_k(d(i)_k)$ be the minimum average dissimilarity from i to other clusters, then the silhouette width of an observation i is defined to be

$$s(i) = \frac{b(i) - a(i)}{\max(a(i), b(i))}. \tag{1.6}$$

The silhouette width, $s(i)$, can obtain values from -1 to 1 corresponding to the state of observation i in its cluster. The recommended interpretation is that if the silhouette width is between 0 and 1 it is "happiest" in its existing cluster; if it is 0, the observation is ambivalent about cluster membership vs. the next closest cluster(s); and if it is between -1 and 0, the observation "wants to leave" the current cluster.

A global measure for a cluster solution is found by averaging all n silhouette widths, defining the average silhouette width as

$$\bar{s} = \frac{\sum_{i=1}^{n} s(i)}{n}. \tag{1.7}$$

The cluster structure with K clusters that has the maximum average silhouette width will be considered as the "optimal" structure. Rousseeuw (1987) suggested that a graphical display of the average silhouette width should also be used to detect unusual average silhouette width values due to outliers.

Although average silhouette width is explicitly recommended by Kaufman and Rousseeuw (1990) for selecting the number of clusters in their PAM algorithm, it can be applied to any cluster solution if the dissimilarity matrix and cluster memberships are available. While average silhouette width is a clear criterion for choosing the number of clusters in a clustering problem, it has a major limitation in that it cannot select a single cluster solution because it is not defined on $K = 1$. In practice, large average silhouette width values for $K = 2$ are often observed when no real clusters exist in the data set, making its use for selecting between $K = 1$ or $K = 2$ problematic.

1.3.2 Caliński and Harabasz (CH)'s Pseudo-F

Caliński and Harabasz (1974) proposed the use of the idea of an F-statistic as a criterion to choose the number of clusters, K, to maximize the variation between clusters relative to the variation within clusters. Their pseudo-F can be calculated as

$$\text{pseudo}-F = \frac{B(K)/(K-1)}{W(K)/(n-K)} \tag{1.8}$$

where $B(K)$ is the between cluster sums of squares

$$B(K) = \sum_{k=1}^{K} \sum_{q=1}^{Q} n_k (\overline{y}_{\cdot qk} - \overline{y}_{\cdot q \cdot})^2,$$

and $W(K)$ is the within cluster sums of squares

$$W(K) = \sum_{k=1}^{K} \sum_{i \in C_k} \sum_{q=1}^{Q} (y_{iqk} - \overline{y}_{\cdot qk})^2,$$

where n_k is the number of data points in cluster C_k, y_{iqk} is the ith observation of variable q in cluster C_k, $\overline{y}_{\cdot qk}$ is the average value for all observations for variable q in cluster C_k, and $\overline{y}_{\cdot q\cdot}$ is the grand mean of all n observations in variable q.

Because the pseudo-F is the ratio of the variance between the groups to the variance in the residuals, the use of K clusters is suggested when the observations are similar within groups (small $W(K)$) but different between groups (large $B(K)$). However, like the average silhouette metric, the pseudo-F needs at least two clusters to be calculated so it cannot select a single cluster solution and often shows large values for $K = 2$ when only one cluster is present.

To overcome the limitation of not being able to consider a one cluster solution (that no clustering should be done), we suggest two methods that are inspired by the decision trees of monothetic clustering to assist in choosing a potentially reasonable number of clusters when no other information about the context or problem is available. Having been widely used in classification and regression trees (Breiman et al., 1984), M-fold cross-validation is adapted for use in deciding the number of clusters in monothetic clustering. Another method is inspired by conditional inference trees (Hothorn et al., 2006), where a hypothesis test is done at every split to assess evidence against the global null hypothesis of independence between the response variable and the covariates. A simulation study is done to compare the performance of suggested methods with average silhouette width and CH's pseudo-F in correctly picking the true simulated number of clusters. Finally, a hybrid method is also suggested to exploit the combined benefits of different methods. More details are given in Chapter 2.

1.4 Clustering Data with Circular Variables

In some applications, a variable can be measured in angles, when the directions of an object or event are observed. Examples include the wind directions in a study interested in counting the number of particles carried in the air in Antarctica (Šabacká et al., 2012), or the aspect of the slope at various positions in a study of the relationship between the snow density with other factors in a mountainous area (Wetlaufer et al., 2016). Such variables are referred to as *circular variables*.

There are several characteristics of circular variables that make them different from conventional variables. Directional data are appropriately measured in angular units (either degrees or radians), creating values that are bound between the "starting point" (e.g., 0 degrees) and the "ending point" (e.g., 360 degrees) where these two points coincide with each other. As the result, these data are usually depicted by angles in a circle, hence the usage of the name *circular variable*. The rotation direction is also an important feature of this type of variable as conventions differ across fields and software. For instance, in the R package **circular** (Agostinelli and Lund, 2017), true east is 0^o and counter-clockwise is the rotation direction, while geographic information system (GIS) software such as ArcGIS (Esri, 2018) takes true north as zero and clockwise as the rotation direction. This means that in R an angle of 60^o would be 30^o in ArcGIS. A related issue is whether the angle is from or to the direction of, say, the wind. It is important for interpretation of results for these aspects to be clearly defined.

To properly display these characteristics, circular variables are usually depicted in or on circles, with the data points put on the circle, or by vectors based on the angles in the circle. Standard exploratory data analysis plots, such as the dotplot, histogram, stem and leaf, density plot, etc., have circular versions. A unique plot that

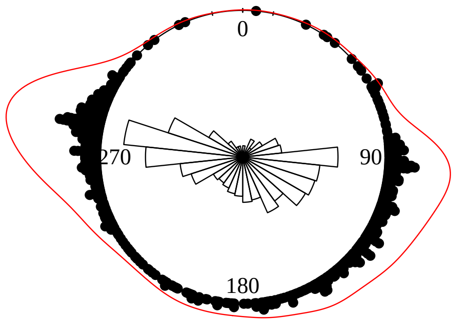


Figure 1.8: A circular data display of the distribution of the wind directions measured at the Bonney Riegel location in McMurdo Dry Valleys in the Antarctic from July 7, 2008 to July 14, 2008 (details in Chapter 3). The plot includes rose diagram (bars), dot plot (dots around circle), and nonparametric density plot (red line outside or on circle).

can be used to examine the distribution of a circular variable is the *rose diagram*, which is believed to first be used in 1858 to depict the effect of sanitation of the hospitals for the British Army by months across the year (the months were treated as circular) (Fisher, 1993). It is very similar to a circular version of the histogram, but the bars are wider at the end. Figure 1.8 is an example of a combination of rose diagram, circular dotplot, and circular density plot on a circle for wind direction data measured in Antarctica (more in Chapter 3).

The analysis of data sets including circular variables with such unique characteristics requires a different set of statistical methods from conventional "linear" variables. There are books dedicated to this topic (e.g., Fisher, 1993; Jammalamadaka and SenGupta, 2001; Pewsey et al., 2013) that develop parametric models and analytic tools for circular variables. They addressed statistical techniques focused

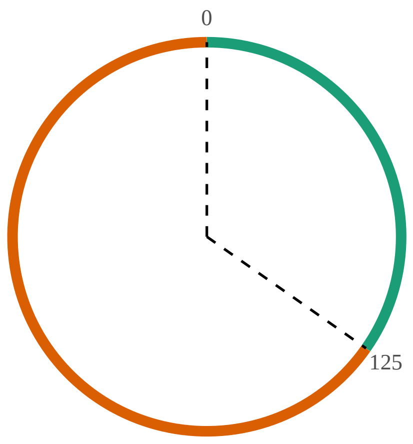
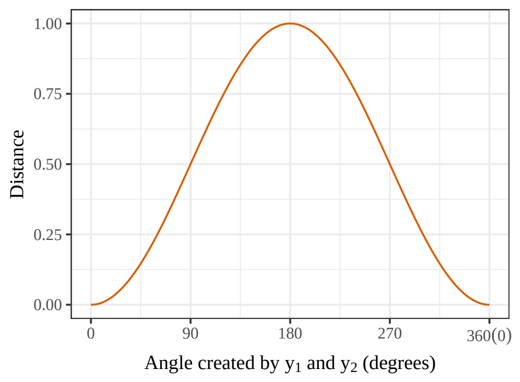


Figure 1.9: An example of a two partition split of a circular variable. Two arcs are created by splits at 0 and 125 degrees.

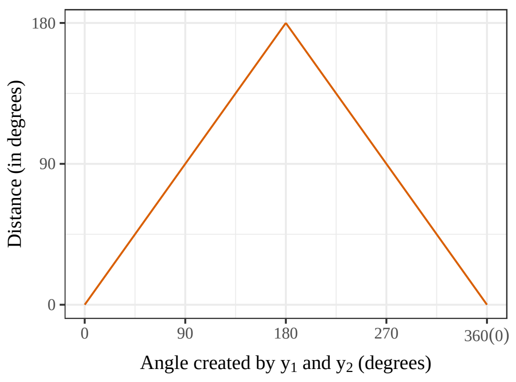
on modeling with circular variables but do not address the topic of multivariate data analysis, specifically classification, regression trees, or visualizations with more than 2 variables.

Circular variables were also mentioned in the context of a regression tree scheme where they are used as response as well as explanatory variables. When a circular predictor is used to split the tree, all of the possible "arcs" formed by two values are examined to determine which one optimizes the target function. Therefore, the split on this variable, if it happens, will rely on two values, not just one value as with a typical quantitative variable (Lund, 2002). For example, in Figure 1.9, two arcs (0, 125) and (125, 0) can only be created by splitting at both 0 and 125.

As mentioned above, the first crucial decision in a clustering problem is how the distance between data points is measured. While there are plenty of available distance metrics for conventional quantitative, categorical, and mixed variables to choose from, there is little literature on this for circular variables. It is important to note that the



(a) Lund's distance.



(b) Smaller arc distance.

Figure 1.10: Distance between two points (in degrees) of a circular variable.

distance metric needs to work in data sets with a mixture of conventional and circular variables.

Lund (1999), while examining the least squares regression for a model involving a circular response and/or predictor, defined the distance between two data points y_1 and y_2 (measured in either radian/degree units) as

$$d_{lund}(y_1, y_2) = \frac{1}{2}\left(1 - \cos(y_1 - y_2)\right). \tag{1.9}$$

The value $d_{lund}(y_1, y_2)$ therefore has the range from 0 when $y_1 = y_2$ and 1 when y_1 is opposite of y_2 in a circle (Figure 1.10a). It is a distance metric because it satisfies the set of four required properties of a distance metric (Arkhangel'skii and Pontryagin, 1990), including:

1. Non-negativity: $d(y_1, y_2) \geq 0$,
2. Identity of indiscernibles: $d(y_1, y_2) = 0 \Leftrightarrow y_1 = y_2$,
3. Symmetry: $d(y_1, y_2) = d(y_2, y_1)$, and

4. Triangle inequality: $d(y_1, y_3) \leq d(y_1, y_2) + d(y_2, y_3)$.

Another possible measure of distance between two data points is the measure of the smaller angle between the points on a circle, i.e.,

$$d_{sma}(y_1, y_2) = 180 - |180 - |y_1 - y_2||\,, \tag{1.10}$$

where y_1 and y_2 are measured in degrees (Jammalamadaka and SenGupta, 2001, Section 1.3.2). When y_1 and y_2 are measured in radians, Eq. 1.10 uses π in place of 180. This measure of distance retains linearity so the change in distance is constant across all values (Figure 1.10b). This distance measure also satisfies all properties of a distance metric and it can be used in monothetic clustering using Equation 1.3.

A (dis)similarity measure for mixed data types is Gower's dissimilarity (Cuadras and Arenas, 1990; Gower, 1971; Pavoine et al., 2009). Gower (1971) proposed a similarity measure among observations from various types of variables: quantitative, categorical, and binary. It can be used in Equation 1.3 when converted to a dissimilarity measure. Gower's dissimilarity for a data set with Q variables is

$$d_{gow}(i, j) = \frac{\sum_{q=1}^{Q} w(y_{iq}, y_{jq}) d(y_{iq}, y_{jq})}{\sum_{q=1}^{Q} w(y_{iq}, y_{jq})}, \tag{1.11}$$

where $d(y_{iq}, y_{jq})$ is the distance between observations i and j regarding the variable q and $w(y_{iq}, y_{jq})$ is the weight coefficient. It can incorporate categorical variables with $d(y_{iq}, y_{jq})$ is 0 if the two observations belong to the same category of q and 1 otherwise. An additional feature of Gower's dissimilarity is that it can also accommodate some missing values, with $w(y_{iq}, y_{jq})$ set to 0 if the value is missing for either or both of the observations and $w(y_{iq}, y_{jq})$ is set to 1 if both are available. This drops the comparison between two observations on that variable and re-weights the resulting dissimilarity.

If q is a linear quantitative variable,

$$d(y_{iq}, y_{jq}) = \frac{|y_{iq} - y_{jq}|}{\max_{i,j} |y_{iq} - y_{jq}|}$$

is a scaled Manhattan (city block) distance, which is 0 when y_{iq} and y_{jq} are the same and 1 for maximum differences. Gower's dissimilarity formula provides distances between 0 and 1. Details and examples can be seen in Everitt et al. (2011). The original Gower's dissimilarity does not have a formula for calculating the distances between pairs of data points in circular variables, but Pavoine et al. (2009) extended Gower's dissimilarity to this type of variable by using Lund's distance, $d_{lund}(y_1, y_2)$, or a fraction of smaller arc distance, $d_{sma}(y_1, y_2)/180$, to ensure that the distance lies in [0, 1] as other Gower's dissimilarity formulas. In Chapter 3, we apply this Gower's dissimilarity for circular variables to achieve a distance matrix of the data set of particle counts in Antarctica which includes a circular wind direction variable and perform monothetic clustering on it to explore structure in that data set.

1.5 Functional Data

Measurements y taken over some ordered index t such as time, frequency, or space and thought of as curves or functions of the index t are called *functional data*, $y(t)$ (Ramsay and Silverman, 2005). Functional data have, possibly, a high frequency of observations over the index t and are assumed to be generated by a smooth underlying process. Some examples of functional data include the growth curves for girls in the Berkeley Growth Study (Tuddenham and Snyder, 1954), hydraulic gradients in wetlands (Greenwood et al., 2011), or daily sea ice extent area over time in the Arctic Sea (Fetterer et al., 2018). The last example is the motivational data set for this dissertation in Chapter 4, although this is the first work to consider these data as functional.

Clustering can be useful for functional data to find groups of curves sharing common characteristics and find representative curves corresponding to different modes of variation in the data. Early applications of clustering functional data considered the functions at arbitrary, or originally observed, discrete values of the argument t, which does not fully capitalize on the functional nature of the data. More recently, various approaches to clustering functional data have been proposed. An extensive overview of various functional data clustering research can be seen in Hitchcock and Greenwood (2015).

Functional data are usually represented as a combination of basis functions and estimated coefficients for each basis function. Tarpey and Kinateder (2003) used k-means on these basis representations of functional data which ends up being a cluster analysis of the estimated basis coefficients when the bases used are orthogonal. In the same paper, Tarpey and Kinateder also showed that if the variability of shapes of curves is of interest, the clustering algorithm can be performed on the first derivative of the function to effectively remove the variability of the mean of each curve.

Some researchers have attempted to adapt traditional multivariate clustering methods to functional data. Distance-based clustering techniques such as agglomerative hierarchical clustering (Clarkson et al., 2002) and PAM (Hitchcock et al., 2007) have been used on dissimilarity matrices from pairs of functions. Tarpey (2007) used L_2 distance between functions, $y_i(t)$ and $y_j(t)$, with t the index within a fixed interval T, defined as

$$d(y_i, y_j) = \sqrt{\int_T [y_i(t) y_j(t)]^2 dt}, \tag{1.12}$$

as a distance measure between functions and their cluster mean in the k-means clustering framework. He showed that using different basis functions (hence different linear transformations) to represent functions can result in very different cluster results. Therefore, this approach is more effective when some linear transformation is performed on the curves to minimize the within-cluster variance and maximize the between-cluster variance. Moreover, when the basis functions are orthogonal, clustering directly on functions using the L_2 metric is equivalent to clustering the raw data using Euclidean distance. Suarez and Ghosal (2016) suggested calculating dissimilarity based on the average number of shared detail coefficients in the Wavelet basis representation of the functions. This approach involves using assigning Dirichlet process prior distributions to the detail coefficients and estimating them by the Monte Carlo Markov Chain (MCMC) method. Model-based clustering (Banfield and Raftery, 1993) is also used in functional data clustering. James and Sugar (2003) further parameterized the function's basis coefficients to form a random effects model to cluster sparsely sampled functional data. Other functional data clustering approaches involve using a conventional clustering approach using rank-based correlation as the dissimilarity measure for functional data (Heckman and Zamar, 2000); or projecting sparsely measured functional data into a reduced-dimensional subspace by applying the functional principal component analysis (FPCA) and then applying conventional clustering techniques on the FPCA scores (Zajacova et al., 2015).

In Chapter 4, we apply monothetic clustering and a similar clustering method that utilizes the combined information of variables in intervals of the functional data called *Partitioning Using Local Subregions* (PULS) on the Arctic sea ice extent data to show that cluster analysis can give some similar conclusions as other studies and suggest new ones. It gives us the ability to characterize similarities and differences in the yearly ice patterns from 1978 to 2018. Also, a simulation study with simulated functional data explores the performance of various clustering techniques in recreating the true cluster membership.

Two criteria are used to evaluate the performance of the clustering methods. The Rand index (Rand, 1971) is a simple measure to compare two different cluster solutions. It counts the number of pairs of observations that either appear together or are separate in two cluster solutions (number of agreements, A), divided by the possible number of pairs from the data set (total number of agreements A and disagreements D),

$$c(C_k, C_{k'}) = A/(A + D) \tag{1.13}$$

Therefore, a higher Rand index indicates a better agreement between two cluster solutions and 1 implies identical solutions. Hubert and Arabic (1985) argued that the original Rand index does not account for "the agreement by chance", which means that there is no common baseline for the "worst case" and hence it is difficult to say which cluster result is better than another based solely on the Rand index. Let n_{ij} be the number of observations that are common to cluster C_i of the cluster result U and cluster C_j of the cluster result V, and $n_{i\cdot}$ and $n_{\cdot j}$ be the number of observations in cluster C_i and C_j, respectively. They suggested an *adjusted* version of the index by subtracting both the numerator and denominator by the expected number of pairs in which the objects are placed in the same clusters in two clustering results

$$c(C_k, C_{k'}) = (A - E)/(A + D - E), \tag{1.14}$$

where

$$E = E\left(\sum_{i,j}\binom{n_{ij}}{2}\right) = \sum_{i}\binom{n_{i\cdot}}{2}\sum_{j}\binom{n_{\cdot j}}{2}\bigg/\binom{n}{2}, \tag{1.15}$$

thus, ensuring that the adjusted Rand index is bounded between 0 and 1. This index is a better measure of agreement between two cluster results if the sizes of the clusters are not uniform or there are many clusters and provides the amount of agreement over chance.

The variation around the cluster means is another criterion used in the simulation study in Chapter 4 to compare the performance of clustering techniques. It is similar to the idea of the coefficient of determination (the proportion of the variance of the response variable explained by a model, R^2). Chavent et al. (2007) used this criterion under the name of *the proportion of inertia explained*, which is the proportion of total within cluster inertia $\sum_k I(C_k)$ (Equation 1.2) over the total inertia of the data set Ω, $I(\Omega)$. This criterion has been subject to some recent criticisms based on the potential to be inflated by scaling or projecting the data (Loperfido and Tarpey, 2018). Even with its potential issues across transformations of data, it provides for somewhat interesting comparisons of different methods on the same distance matrix.

1.6 R Implementation of the Suggested Methods

Most of the statistical methods mentioned above have been implemented in various R (R Core Team, 2019) packages. Here is the overview of some packages used in this dissertation.

Commonly used clustering methods such as k-means and Ward's hierarchical clustering are implemented in functions `kmeans` and `hclust`, respectively, in the core **stats** package. There are two realizations of Ward's method in the `hclust` function, defined in the argument `method` of that function. *Ward1* is implemented in the `ward.D` option while *Ward2* is achieved by using the `ward.D2` option. Chavent et al. (2007) created a package for their DIVCLUS-T method called **divclust** on GitHub (`https://github.com/chavent/divclust`). **divclust** can perform monothetic clustering on numerical and/or categorical variables, visualize the splitting rules tree, and calculate the total inertia at each step to assist the choice of number of clusters.

However, because it is not based on the classification and regression tree (CART) implementation (although the method is inspired by that method), its implementation lacks several options such as specifying the minimum number of observations that must exist in a node in order for a split to be attempted (minsplit) or the minimum number of observations allowed in a terminal leaf (minbucket) and is not compatible with existing output and plots commonly used in the **rpart** package (Therneau and Atkinson, 2018).

The **rpart** package is an R package that implements methods in *"Classification and Regression Trees"* (Breiman et al., 1984). The package focuses on creating regression trees with text and graphical outputs, implementing cross-validation methods to prune the trees and dealing with missing data in the CART methodology. It is considered as a "standard" package for CART problems and the basis of other packages such as **party** (Hothorn et al., 2006) created to improve the visualizations of **rpart** results.

There are several packages in R that are designed to work with circular or directional data. Some packages focus on the visualization of individual circular variables (**circlize**, Gu et al., 2014) while other packages are designed for specific data sources such as circularly distributed stimulus variables in psychology (**CircularDDM**, Lin et al., 2018) or non-parametric circular methods (**OmicCircos**, Oliveira et al., 2014). **circular** (Agostinelli and Lund, 2017) is another package that has many useful functions related to circular variables, such as standard plots and model fitting with circular variables. Pewsey et al. (2013) discussed the use of this package in their book related to circular statistics. However, as far as we know, there is no package that is designed to perform clustering on variables with circular variables. There are packages designed for making PCPs with categorical variables such as **ggalluvial** (Brunson, 2018) or **ggparallel** (Hofmann and Vendettuoli, 2013), extensions of existing functions in **ggplot2** (Wickham, 2016). However, there are no packages that can mix circular variables into PCPs, as discussed in Chapter 3.

A commonly used R package for performing FDA is the **fda** package (Ramsay et al., 2018) which was developed to accompany the classic FDA book *"Functional Data Analysis"* by Ramsay and Silverman (2005). Febrero-Bande and Fuente (2012) extensively added other FDA methods such as depth measurements, regression models, and unsupervised classification, among others, to the **fda** package with the **fda.usc** package.

In this dissertation, most of the methods and functions are implemented in the **monoClust** package maintained on GitHub (`https://github.com/vinhtantran/monoClust`). This package was started by Brian McGuire and Mark Greenwood as a small project and transferred to the current authors to maintain and improve. This package reuses and modifies several core functions and arguments of the **rpart** package, especially the text and plot outputs. It implements monothetic clustering on numerical, categorical (in experimental phase), and circular variables. It can also perform clustering on functional data in their discretized forms. Its extended functionalities include plotting splitting rule trees in more details and colors, assisting the choice of number of clusters by a permutation-based hypothesis test, and making PCPs plot with circular variables. Additionally, another clustering method inspired by monothetic clustering but designed to work directly with functional data called *Partitioning Using Local Subregions* (details in Chapter 4) has also been assembled in a separate package named **PULS** on GitHub (`https://github.com/vinhtantran/PULS`). The functionalities, arguments, and usage of the two packages are described in detail in Chapter 5.

CHAPTER TWO

CHOOSING THE NUMBER OF CLUSTERS IN MONOTHETIC CLUSTER ANALYSIS

Contribution of Authors and Co-Authors

Author: Tan V. Tran

Contributions: Responsible for the majority of the writing.

Co-Author: Mark C. Greenwood

Contributions: Provided guidance and feedback on statistical analysis and drafts of the manuscript.

Manuscript Information

Tan V. Tran and Mark C. Greenwood

Electronic Journal of Applied Statistical Analysis

Status of Manuscript:

__X__ Prepared for submission to a peer-reviewed journal

_____ Officially submitted to a peer-reviewed journal

_____ Accepted by a peer-reviewed journal

_____ Published in a peer-reviewed journal

Published by Università del Salento

Abstract: Monothetic clustering is a divisive clustering method based on recursive bipartitions of the data set determined by choosing splitting rules from any of the variables to conditionally optimally partition the multivariate responses. Similar to other clustering methods, the choice of the number of clusters is important in this method to facilitate interpretations when there is no a priori *information about a reasonable number of clusters. The connections between monothetic clustering and decision trees motivate the consideration of pruning methods to aid in the selection of the number of clusters. We propose a cross-validation technique to find the number of clusters that optimize prediction error or using permutation-based hypothesis tests at each bi-splitting step, retaining splits with "small" p-values. A simulation study is performed to evaluate the performance of the new methods and compare to some other existing techniques.*

2.1 Introduction

Cluster analysis (or clustering) attempts to group observations into clusters so that the observations within a cluster are similar to each other and dissimilar from observations in other clusters. It is often used when dealing with the question of discovering structure from data where no supervised variable, the variable used to evaluate the structure of other variables, exists. Therefore, cluster analysis is considered a type of unsupervised learning. It is used in many fields including statistics, machine learning, and image analysis. For a general introduction to cluster analysis, see Everitt and Hothorn (2011, Chapter 6).

Let y_{iq} be the i^{th} observation ($i = 1, \ldots, n$, where n is the number of observations or sample size) on variable q ($q = 1, \ldots, Q$, the number of response variables). Because there is no supervised variable, all Q variables are considered "response" variables and the interest is in exploring potential groups in these (multivariate) responses.

Occasionally other information in the data set is withheld to be able to understand clusters found in Q variables used in clustering. Clustering algorithms then attempt to partition the n observations into K mutually exclusive clusters $C_1, C_2, \ldots, C_K$, so that the observations within a cluster are “close” to each other and “far away” from those in other clusters. Clustering algorithms depend on the choice of data, the desired number of clusters, and the selected measure of distance or dissimilarity among observations. Commonly used distance metrics are Euclidean distance, Manhattan distance, and Gower’s distance but there are many other ways to define proximity of multivariate observations.

Because of the typically unknown structure of the data, there is no known “correct” solution in most clustering problems. In general, it is desirable for a clustering algorithm to create interpretable clusters, be computationally efficient, and identify true structure in data sets, when it exists. Given a clustering algorithm, the ability to interpret clusters is heavily influenced by the choice of K, the number of clusters. If K is too small, it puts “dissimilar” observations together. On the other hand, if K is too large, it would split observations into different clusters that share many characteristics. Therefore, picking a “sufficient” K is critical for any clustering algorithm. Moreover, there is also a question of whether any cluster structure is present in the data for clustering to even take place. When the data do not have a true cluster structure (e.g., a data set generated from a single Gaussian distribution), the results of a clustering algorithm can be arbitrary and misleading. This problem is related to the quantification of the degree of cluster structure, called *notions of clusterability* (Ben-David, 2015). In a paper detailing the question of one vs. more than one cluster, Adolfsson et al. (2019) suggested several *measures of clusterability*, a measure of the degree of inherent cluster structure, based on dimension reduction techniques (e.g., principal component analysis) and multimodality tests (e.g., Hartigan's dip test, Hartigan and Hartigan, 1985). To compare measures of clusterability, they performed a simulation study using different cluster structures, such as one true cluster to assess the Type I error results for unclusterable data and more than one true cluster to evaluate the power of the measures, among others. These methods were developed to be independent of the choice of clustering algorithm but do depend on choices of variables and dissimilarities.

Here, we first describe the use of monothetic cluster analysis in Section 2.2. Next, we give an overview of several well-known methods for choosing the number of clusters in a cluster analysis (Section 2.3) and introduce two novel methods that use cross-validation and permutation tests to pick the "reasonable" number of clusters (Section 2.4), which also relates to a type of clusterability assessment in monothetic clustering. Finally, in Section 2.5, a comparison among those methods is performed using a simulation study. Conclusions are presented in Section 2.6.

2.2 Monothetic Cluster Analysis

Monothetic clustering is a type of clustering that provides a hierarchical, recursive partitioning of multivariate responses based on binary decision rules that are built from individual response variables (Sneath and Sokal, 1973). Chavent (1998) developed an algorithm implementing *monothetic cluster analysis* that was inspired by classification and regression trees (CART, Breiman et al., 1984). This monothetic clustering algorithm searches for splits from each response variable that provide the best split of the multivariate responses in terms of a global criterion called *inertia*, the total variability around the cluster centroid, and recursively partitions the data set. In the case of Euclidean distances, the inertia, $I(C_k)$ for cluster C_k, is defined as

$$I(C_k) = \sum_{i \in C_k} d^2_{euc}(\mathbf{y_i}, \overline{y}_{C_k}), \tag{2.1}$$

where $\overline{y}_{C_k}$ is the centroid of all observations in cluster C_k and $d^2_{euc}(\mathbf{y_i}, \mathbf{y_j})$ is the Euclidean distance between observations $\mathbf{y_i}$ and $\mathbf{y_j}$, defined as

$$d_{euc}(\mathbf{y_i}, \mathbf{y_j}) = \sqrt{\sum_q (y_{iq} - y_{jq})^2}. \tag{2.2}$$

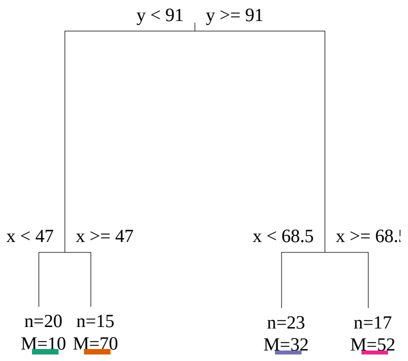
It has been shown that the sum of squared Euclidean distances for all observations within a cluster and variation around the means within a cluster are equivalent (James et al., 2013, p 388), so the within cluster inertia (Equation 2.1) can be equivalently calculated from the dissimilarity matrix as

$$I(C_k) = \frac{1}{n_k} \sum_{(i,j) \in C_k, i>j} d^2_{euc}(\mathbf{y_i}, \mathbf{y_j}). \tag{2.3}$$

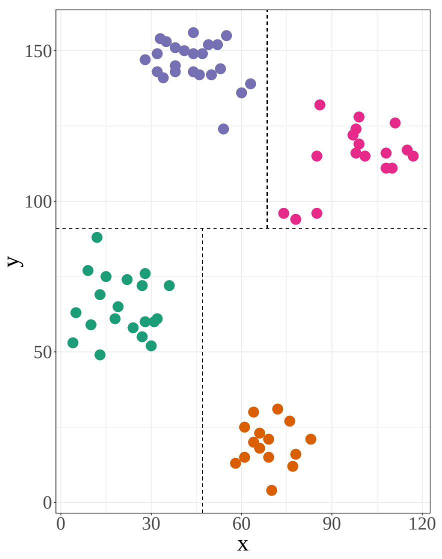
This result is used to justify the application of many distance-based methods to non-Euclidean dissimilarities (Anderson, 2001). Hence the monothetic clustering algorithm can also be directly applied to non-Euclidean distances and other dissimilarities, including data that contain circular variables and mixed data with categorical and quantitative variables or even data with missing observations.

A binary split $s(C_k)$ on a cluster C_k divides its observations into two smaller clusters C_{kL} and C_{kR}. The inertia decrease based on a proposed partition is defined as

$$\Delta(s, C_k) = I(C_k) - I(C_{kL}) - I(C_{kR}), \tag{2.4}$$



(a) Tree of splitting rules.



(b) Cluster result with splitting rules.

Figure 2.1: Monothetic clustering on the *ruspini* data with four cluster solution.

and the best split $s^*(C_k)$ is the split that maximizes this decrease in inertia,

$$s^*(C_k) = \arg\max_s \Delta(s, C_k). \tag{2.5}$$

The same algorithm is then recursively applied to each sub-partition, recording splitting rules on its way. The result of the algorithm is a set of hierarchical binary rules for determining cluster membership. The resulting hierarchy can be read and displayed as a decision tree.

Figure 2.1 shows an example of the splitting rules and the created clusters for the *ruspini* data set (Ruspini, 1970) using monothetic clustering for the $Q = 2$ variables (x and y) and $n = 75$ observations. The monothetic clustering algorithm suggests the first split at $y = 91$. In the newly created cluster that includes the data points that had $y \geq 91$, the algorithm suggests splitting at $x = 68.5$, while for observations with $y < 91$, it suggests partitioning at $x = 47$. The set of splitting rules can be summarized in hierarchical tree form as shown in Figure 2.1a with each rule being generated from one variable. The rules can be used to generate an interpretation of the four selected clusters as: Cluster 1 includes observations that have $y < 91$ and $x < 47$, while observations having $y < 91$ and $x > 47$ belong to cluster 2. Similarly, having $y > 91$ and $x < 68.5$ are the characteristics of the observations in cluster 3, and the rest of the data set that has large y (> 91) and x (> 47) belongs to cluster 4. The clusters are visualized in Figure 2.1b. The plot colors are coded to be consistent with the color in the terminal nodes of the tree. Using the `MonoClust` function (discussed in Chapter 5), the splitting rules can also be printed as follows.

```
n= 75

Node, N, Within Cluster Deviance, Proportion Deviance Explained,
      * denotes terminal node

1) root 75 250000 0
  2) y < 91 35  43000 0.63
    4) x < 47 20   3700 0.16 *
    5) x >= 47 15   1500 0.16 *
  3) y >= 91 40  47000 0.63
    6) x < 68.5 23   3200 0.16 *
    7) x >= 68.5 17   4600 0.16 *
```

While Chavent (1998) used simple data sets, *ruspini* and *iris* (Fisher, 1936), to demonstrate the monothetic clustering algorithm, Piccarreta and Billari (2007) have also used the same monothetic divisive clustering idea to group women based on their life course states (marriage, child-bearing, and having a job) in each year of their lives, and creating groups of life course trajectories based on the combination of work and family orientation. Although they decided to retain 12 clusters, much more than three original categories from a previous study (family oriented, work oriented, and “the best of both worlds” groups), they are still able to interpret the resulting clusters using the participants' life course states due to the rules defining those clusters.

Chavent et al. (2007) compared monothetic clustering to k-means (MacQueen, 1967) and Ward's method (Ward, 1963) by simulation and application to six real-life data sets (three data sets with numerical variables and three data sets with categorical variables) from the UCI Machine Learning repository (Dua and Graff, 2019). Compared to k-means and Wards' method, a benefit of monothetic clustering is the ability to interpret the clusters based on shared characteristics from the split variables and predict new observations by using the decision tree. However, the partitioning based on one variable at a time is not as flexible as some other algorithms, limiting its performance for complicated data structures and cluster shapes. They found that monothetic clustering performed better than k-means and Ward's methods in terms of correct classification of known group memberships when the number of clusters is small. Also, for numerical data sets, the larger the data set is, the better result monothetic clustering had. Nevertheless, there have been limited studies to compare monothetic clustering performance to other algorithms.

2.3 Some Existing Metrics for Choosing the Number of Clusters

Deciding on the number of clusters to report and interpret is an important part of cluster analysis. In many applications, researchers can decide on a sufficient number of groups based on the knowledge of the subject or sometimes based on visualization of the data or clusters. However, in many cases, when there is little knowledge about the structure of the data, a criterion based on the cluster results themselves may provide useful information for this problem. Many metrics for choosing the number of clusters have been mentioned in the clustering literature, such as papers on the comparison of metrics by Milligan and Cooper (1985) and Hardy (1996), and also implemented in statistical software such as the package **NbClust** (Charrad et al.,

2014) in R (R Core Team, 2019). In those papers, Caliński and Harabasz (CH)'s pseudo-F (Caliński and Harabasz, 1974) is among the metrics that have typically good or even the best performance in the simulation studies on selecting the optimal number of clusters, depending on the scenario and algorithm considered. Average silhouette width (Rousseeuw, 1987) is a measure of how "comfortable" an observation is in the cluster it resides in and has been commonly used to select an appropriate number of clusters, especially for applications of the Partitioning Around Medoids (PAM) algorithm (Handl et al., 2005; Kaufman and Rousseeuw, 1990).

2.3.1 Average Silhouette Width

Silhouette width (Rousseeuw, 1987) is a measure of how "comfortable" an observation is in the cluster it resides in. Let $a(i)$ be the average dissimilarity between observation i and other observations in the same cluster, $d(i)_k$ be the average dissimilarity between i and other observations in cluster k, and $b(i) = \min_k(d(i)_k)$ be the minimum "distance" from i to other clusters, then the silhouette width is defined to be

$$s(i) = \frac{b(i) - a(i)}{\max(a(i), b(i))}. \tag{2.6}$$

The silhouette value for the i-th observation, $s(i)$, can then obtain values from -1 to 1. The recommended interpretation is that if the silhouette width is between 0 and 1, the observation is "happiest" in its existing cluster; if it is 0, the observation is ambivalent about cluster membership versus next closest cluster; and if it is between -1 and 0, the observation "wants to leave" the current cluster.

A global measure for a cluster solution is found by averaging all n silhouette widths, defining the *average silhouette width* for a K cluster solution as

$$\overline{s_K} = \frac{\sum_{i=1}^{n} s(i)}{n}. \tag{2.7}$$

The cluster structure that has the maximum average silhouette width is considered as the "optimal" structure.

Average silhouette width is recommended by Kaufman and Rousseeuw (1990) for selecting the number of clusters specifically in their PAM algorithm, but it can be applied to any cluster solution if the dissimilarity matrix and cluster memberships are available. Although average silhouette width is a clear criterion for choosing the number of clusters in a clustering problem, it has a major limitation in that it cannot select a single cluster solution because it is not defined on $K = 1$. In practice, large average silhouette width values for $K = 2$ are often observed when no real clusters exist in the data set, making its use for selecting $K = 1$ or $K = 2$ problematic. Moreover, outliers are influential in the average silhouette width calculation. Because of this, Kaufman and Rousseeuw (1990) suggested that unusual observations be removed from the average silhouette width calculation, making subjective decisions about outliers being singleton clusters part of using this criterion in practice.

2.3.2 Caliński and Harabasz (CH)'s Pseudo-F

Caliński and Harabasz (1974) proposed the use of an F-statistic as a criterion to choose the number of clusters, K, in order to maximize the ratio between the variation between clusters and the variation within clusters. The pseudo-F can be calculated as

$$\text{pseudo}-F = \frac{B(K)/(K-1)}{W(K)/(n-K)} \tag{2.8}$$

where $B(K)$ is the between cluster sums of squares

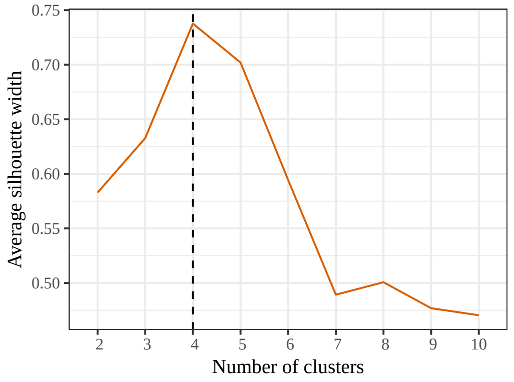
$$B(K) = \sum_{k=1}^{K} \sum_{q=1}^{Q} n_k (\overline{y}_{\cdot qk} - \overline{y}_{\cdot q \cdot})^2,$$

and $W(K)$ is the within cluster sums of squares

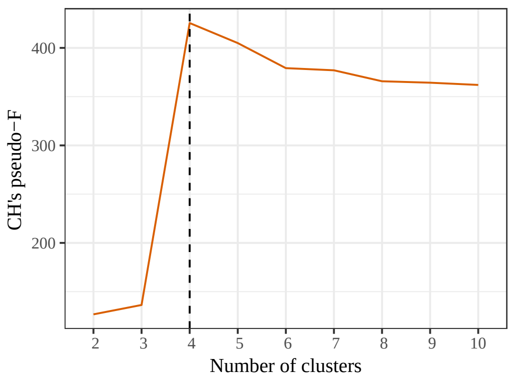
$$W(K) = \sum_{k=1}^{K} \sum_{i \in C_k} \sum_{q=1}^{Q} (y_{iqk} - \overline{y}_{\cdot qk})^2,$$

where n_k is the number of data points in cluster C_k, y_{iqk} is the ith observation of variable q in cluster C_k, $\overline{y}_{\cdot qk}$ is the average value for all observations for variable q in cluster C_k, and $\overline{y}_{\cdot q \cdot}$ is the grand mean of all n observations in variable q. Both $B(K)$ and $W(K)$ can be found from a dissimilarity matrix as a result of Equation 2.3, avoiding the need to calculate $\overline{y}_{\cdot qk}$ or $\overline{y}_{\cdot q \cdot}$. Because the pseudo-$F$ is the ratio of the variance of the groups to the variance in the residuals, K clusters are considered a good choice when the observations are similar within groups (small $W(K)$) but different between groups (large $B(K)$). However, like the average silhouette metric, the pseudo-F needs at least two clusters to be defined so it cannot select a single cluster solution and often shows large values for $K = 2$ when only one cluster is present.

In Figure 2.2, the average silhouette width and CH's pseudo-F methods are applied to the *ruspini* data to find the “optimal” number of clusters with monothetic clustering. In both methods, the criteria agree with each other and reach their maxima at $K = 4$, suggesting the four cluster solution for this data set.



(a) Average silhouette: selects 4 clusters.



(b) CH's pseudo-F: selects 4 clusters.

Figure 2.2: The choice of clusters for the *ruspini* data made by Average silhouette width and CH's pseudo-F methods for monothetic clustering. Both suggest the use of the $K = 4$ cluster solution.

2.4 Proposed Methods for Choosing the Number of Clusters

Average silhouette width and CH's pseudo-F methods have shown their utility for selecting optimal numbers of clusters when the true $K > 1$. However, they are both unable to ever select a single cluster structure because their formula require at least two clusters to be defined. Moreover, there is no "best" method across all cluster analyses. Depending on the shape of the data and/or the clustering algorithm, a specific metric may be better or worse than others in its suggestion for the number of clusters. Because monothetic clustering is inspired by CART (Breiman et al., 1984), methods developed for pruning the trees have potential to select the number of clusters in monothetic cluster analysis. In particular, we explore an adaptation of a cross-validation technique, where a global criterion related to prediction error is used to compare different cluster solutions. Another approach inspired from conditional inference trees (Hothorn et al., 2006) uses hypothesis tests at each split to decide if a split should be performed or not.

2.4.1 M-Fold Cross-Validation

Cross-validation (CV) is a popular method to "tune" many methods (see Hastie et al., 2016, for example). In CART, cross-validation is used to prune the decision tree after it is grown, often into an overly complicated (too large) tree with more terminal nodes (groups) than needed. The M-fold cross-validation randomly partitions data into M subsets with equal (or close to equal) sizes. $M - 1$ subsets are used as the training data set to create a tree with a desired number of terminal nodes and the other subset is used as validation data set to evaluate the predictive performance of the trained tree. The process repeats for each subset as the validating set and the mean squared difference of the observed and predicted is calculated.

To perform cross-validation in monothetic clustering, the partitioning tree is built based on a training data set. An observed value, y_{iq}, of the validation data set will fall into one of the K clusters based on applying the rules from the tree built with the training data and is predicted to be equal to the "central" point of that cluster, $\hat{y}_{(-i)q}$, which is the cluster mean on the variable q of the cluster created by the training data when Euclidean distance is used. The estimate for the test error for the m-th subset is

$$MSE_m = \frac{1}{n_m} \sum_{q=1}^{Q} \sum_{i \in m} d_{euc}^2(y_{iq}, \hat{y}_{(-i)q}). \tag{2.9}$$

This process is repeated for the M subsets of the data set and the average of these test errors is the cross-validation-based estimate of the mean squared error of predicting a new observation. The overall cross-validation based estimate is then

$$CV_K = \overline{MSE} = \frac{1}{M} \sum_{m=1}^{M} MSE_m. \tag{2.10}$$

The purpose of the cross-validation is to find a cluster solution that achieves the smallest prediction error for new observations by comparing CV_K for different K. A naive approach is to pick the solution that has the smallest CV_K (*minCV* rule). However, in many cases, it can result in a very high number of clusters as the error rate often keeps decreasing slightly as the number of clusters grows. To avoid this problem in CART, Breiman et al. (1984) suggested picking the solution that is simplest within 1 or 2 standard errors of the minimum error estimate (*CV1SE* and *CV2SE* rules), where the standard error (SE) is

$$SE = \sqrt{\frac{1}{M-1}\sum_{m=1}^{M}(MSE_m - \overline{MSE})^2} \tag{2.11}$$

and is generated from the cluster solution with the smallest CV_K.

Because monothetic clustering is one of the few clustering algorithms that provides a clear cluster prediction rule, we can use the binary rules to assign a new observation to a cluster and use the multivariate cluster mean, when Euclidean distance is used, to predict the response to find the cross-validation error estimate. Further, because the MSE can be calculated for $K = 1$, the cross-validation method can compare between the solution of one cluster versus more than one cluster. When Euclidean distance is not used, the representative point of a cluster $\hat{y}_{(-i)q}$ can be estimated using the medoid point (Kaufman and Rousseeuw, 1990). However, the downside of the cross-validation methods is the tree has to be created M times for each cluster size K one wants to check. It would be computationally expensive for data sets with large n and/or large Q. Extensions of these ideas to other metrics or dissimilarities have not been developed as far as we know. Simulation studies in Section 2.5 assess its performance and compare it to other methods for Euclidean distances.

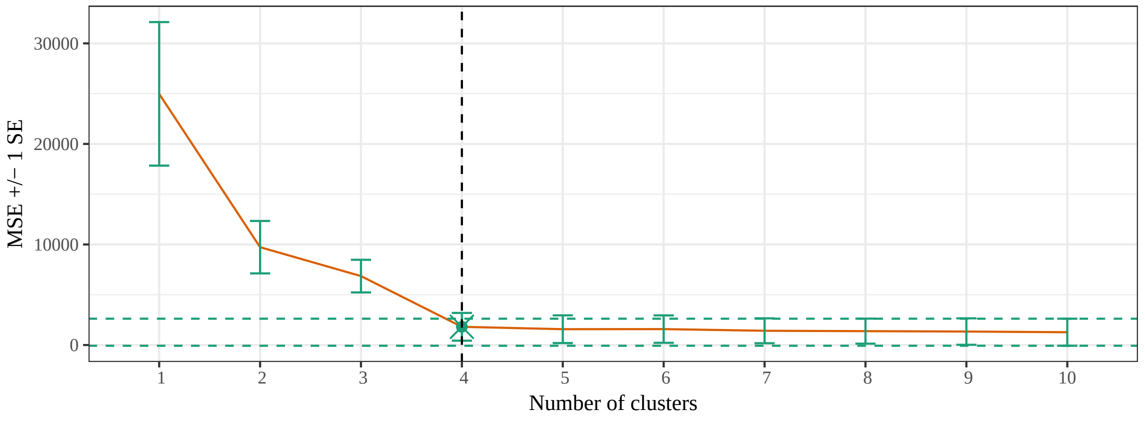


Figure 2.3: The choice of clusters for *ruspini* data made by 10-fold CV where *minCV* selects 10 clusters and *CV1SE* selects 4.

An example of the criterion for the *ruspini* data is in Figure 2.3. The prediction mean squared error (CV_K) decreases when the number of clusters K increases. Ideally, we should see CV_K decrease and then increase, showing a clear choice of K that minimizes prediction error. In this case, the number of clusters explored was not large enough to observe an increase in CV_K, if it occurs. That is the reason why the *minCV* rule suggested that the maximum number of clusters considered as optimal, which was 10. When adding the amount of 1 standard error to this minimum CV_K, it creates a region that covers solutions of 4 to 10 clusters. Because 4 is the smallest cluster solution in the region, it is the solution suggested by the *CV1SE* rule. We can see that the *CV1SE* rule is much more conservative (i.e., choosing fewer clusters) than the *minCV* rule, and it often picks the solution at or near the “elbow” in the CV_K plot.

2.4.2 Hypothesis Tests at Each Bipartition

This section is based on the idea of a formal hypothesis test for each split in the partitioning tree and using p-values to stop tree growth. Specifically, the tree is grown until a split has a p-value higher than a pre-determined threshold (α or Type I error rate) and then that split or any other sub-divisions are not considered. This threshold is continually adjusted when the split is further down the tree to account for inflated Type I error rates as sequences of tests are combined using Bonferroni corrections (discussed below).

For monothetic cluster analysis, at any cluster (or node on the decision tree), the proposed idea is to perform a hypothesis test, in which the hypotheses can be stated as

H_0 : The two new clusters are identical to each other, and

H_A : The two new clusters are different from each other.

If there is strong evidence against the null hypothesis, then the cluster will be split into two new clusters and the algorithm goes on to find the next best split in each cluster. Alternatively, if the test shows insufficient evidence to reject the null hypothesis, the monothetic algorithm stops, no other split is considered, the examined cluster will be the last cluster in the cluster solution, and the tree is the final tree. The pair of hypotheses, when clearly defined below, dictate how the test is done, what statistic to be used, and how evidence against the null hypothesis (p-value) is calculated.

To allow applications with any dissimilarity measure, a nonparametric method based on a permutation test is used. Permutation-based tests are attractive because they do not require distributional assumptions (Berry et al., 2016; Boik, 1987). This property is useful in clustering when the data points can come from any multivariate distribution function. Anderson (2001) developed a multivariate nonparametric testing approach called *perMANOVA* that involves calculating the pseudo-F_{obs}-ratio (same as defined by Caliński and Harabasz, 1974) directly from any symmetric distance or dissimilarity matrix using Equation 2.8 where the sum of squares are calculated directly from the dissimilarities (Equation 2.3). The p-value can then be calculated by tracking the pseudo-F^* across B permutations and comparing the results to our observed result, with p-value $= \frac{\text{count}(F^* \geq F_{obs})}{B}$.

A similar process has been used in the context of conditional inference trees by Hothorn et al. (2006). They used a different test statistic but used its p-values to limit trees developed to explain categorical, quantitative, or multivariate responses.

Some adjustments need to be made to account for the differences with monothetic clustering compared to typical applications of *perMANOVA*. The aim of the permutation test in clustering is to simulate sampling distribution of the test statistic under the null hypothesis that two clusters are identical to each other. Under that null hypothesis, the cluster labels are arbitrary, and can be randomly shuffled to any observation. The pseudo-F's calculated from the shuffles using Equation 2.8 create the reference distribution and the proportions of the shuffled pseudo-F's that are greater than or equal to the observed pseudo-F of the two original clusters out of the number of shuffles, B, is the p-value of the test.

However, a complication that arises in these tests is that in monothetic clustering, the split is based on the variable that has the maximum decrease in the difference inertia between the current cluster and two new proposed clusters (Equation 2.5). This affects the permutation-based hypothesis test by systematically creating p-values that tend to be too small relative to stated Type I error rate because the chosen split is already the most extreme result possible on the variable that defined the bipartition so results in inflated Type I error rates. A possible remedy is to use only variation from the $Q - 1$ variables not used to define the candidate split in the calculation of the pseudo-F statistic. Then the hypothesis test will assess whether the binary split is useful on other variables by assessing the differences between the two groups on the other $Q - 1$ variables. This modification means the test cannot be performed in a data set that has only one variable ($Q = 1$). We call the shuffling between two cluster labels with the splitting variable excluded from the pseudo-F statistic calculation *cluster shuffling*.

Even with the suggested remedy above, the fact that the optimization was not repeated in developing the permutation distribution may be problematic. To avoid that issue, we also propose another permutation-based hypothesis test that uses a slightly different method for generating the null distribution. In this approach, we first find our optimal split on the chosen variable and calculate some "measure of clustering" as an observed test statistic. Then, we permute the values of the selected splitting variable and repeat the optimization of finding an optimal split on that variable to calculate the test statistic. The permuted test statistics form a permutation distribution under the null hypothesis which is used to calculate a p-value based on the number of times the permuted and then optimized results exceed the original optimal result. Two different measures of clustering are considered, which are the average silhouette width (AW) and CH's pseudo-F. We call this permuted hypothesis test method *variable shuffling*.

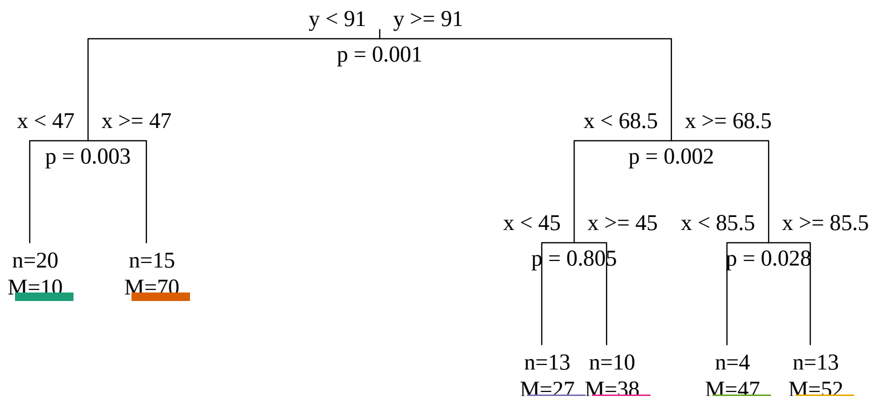
In both *cluster shuffling* and *variable shuffling* approaches, the p-value also needs to be adjusted to account for multiple hypothesis tests required to reach a node further down in the tree. For example, in the *ruspini* data set, the second split happens with x at 68.5 but the hypothesis test for the split is conditional on the result of the hypothesis test at its father node (where y was cut at 91 after rejecting its null hypothesis). The next split at $x = 47$, in turn, is conditional on the rejections of the previous two hypothesis tests. The probability of a Type I error on at least one of these two tests is inflated unless we control for the accumulating number of tests. This probability is getting higher and higher as the tree grows. Bonferroni-adjustment of p-values involves multiplying the p-value by the number of tests required to get to that node to consider it for splitting and can be used to control the overall Type I error rate for each set of tests. Specifically, we use

$$p\text{-value}_{adj} = \min((\text{No. of previous tests} + 1) \times p\text{-value}_{unadj}, 1). \tag{2.12}$$

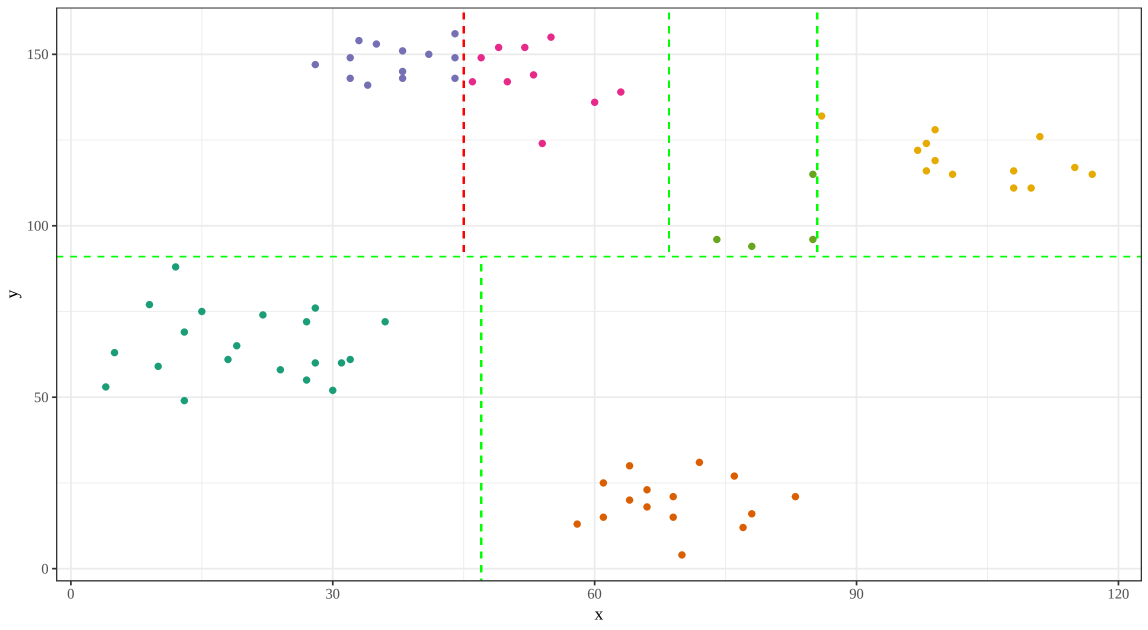
These adjusted p-values are compared to a pre-defined cutoff to decide if a split should be made (p-value$_{adj}$ < cutoff). Note that this is a slightly different use of Bonferroni corrections than Hothorn et al. (2006) considered; they used depth in tree to adjust their p-values.

For example, Figure 2.4 is the monothetic splitting tree with the *cluster shuffling* permutation test's p-value and AW statistic at each node. In the first four partitions, the p-values from the permutation-based hypothesis tests are small (< 0.05), indicating strong evidence that the split should be made. However, at the fifth split (data are split at a x-value of 45 to generate the six cluster result), the test has a very large adjusted p-value ($p = 0.745$), indicating that the split should not be made and the tree growing should stop. Visually, in the cluster which has observations having $y > 91$ and $x < 68.5$, a partition at $x = 45$ looks like a slice in the middle of the points. If we consider the difference between two new partitions in terms of the y-values only, the observations are basically the same. So in this case, the hypothesis testing method suggests the five cluster solution.

The advantage of using these approaches in picking the cluster solution is that it can choose the one cluster solution. The test can also be performed at the root of the decision tree to decide if the data set should be partitioned at all. We assess the "clusterability" using our suggested permutation-based methods in Section 2.5.1. However, permutation-based tests can have a higher rate of wrongly rejecting the null hypothesis when in fact it is true in some situations, such as when there are differing distributions, especially variances, across groups (Boik, 1987). Another advantage of



(a) Splitting rule tree.



(b) Scatterplot of the clusters.

Figure 2.4: Monothetic clustering on the *ruspini* data with permutation-based hypothesis tests (*cluster shuffling* method with pseudo-F statistic) applied at the splits. Panel (a) contains the splitting rule tree with permuted p-value, and panel (b) contains the scatterplot with the cuts as dashed lines. The last split happens at $x = 45$ with $p = 0.745$, so the split (red dashed line) is not performed and the clustering stops at the 5 cluster result.

this approach is the test can be embedded into the tree building procedure, so that the test is performed every time a split is about to be made. With this combination, the cluster analysis algorithm stops when the predefined significance level is crossed. It may depend on the application to assess whether this is an advantage or limitation of this method.

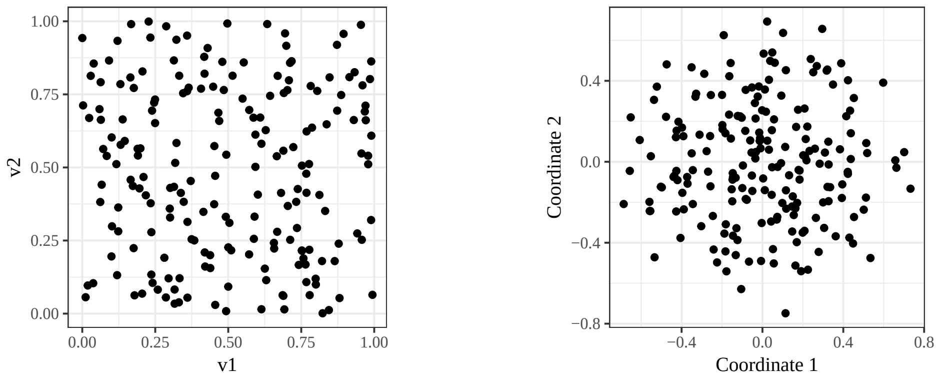
2.5 Simulation Study

2.5.1 Study 1: Unclusterable Data

The first simulation study assesses the ability of the methods in deciding whether some cluster structure (with more than two clusters) is present in the data set or not. Failure to identify data generated from a multivariate uniform distribution as unclusterable leads to misleading cluster results and is a waste of time and resources.

A similar study to evaluate the Type I error of the measures of clusterability has been done by Adolfsson et al. (2019). Type I error in a hypothesis test occurs when the test incorrectly rejects the null hypothesis (Casella and Berger, 2002). Knowing how a hypothesis test controls Type I error rate corresponding to a significance level is helpful to decide whether that test is suitable for an application. A test that rejects the null hypothesis with a higher error rate than nominal is a liberal test and is a dangerous test to use while a test that is conservative may have low power to detect real differences when present.

In order to check the Type I error rate of the suggested permutation tests with different numbers of observations n and variables Q, we created 3 groups of data sets with different sizes, namely 200×4, 200×8, and 300×4. Each group has 1,000 data sets with values randomly sampled from a multivariate uniform distribution in R (R Core Team, 2019). These data sets were designed to have only one true cluster of all data points so no distinct groups are present (Figure 2.5). The proportions of



(a) The first two variables, v1 and v2, in the data set.

(b) The first two coordinates after applying multidimensional scaling to the data set.

Figure 2.5: The realization of a simulated unclusterable data set (200×4). The data set was simulated from a multivariate uniform distribution.

incorrectly choosing a greater than or equal to two cluster result are calculated for the permutation-based hypothesis tests criteria including the cluster shuffling and the variable shuffling approaches (either using AW or F statistics). In each permutation-based hypothesis test, the observed test statistic is compared to the reference null distribution from 1,000 permutations to calculate the p-value. A threshold of $\alpha = 0.05$ is used to decide whether to reject the null hypothesis or not. The cross-validation-based criteria (Section 2.4.1) are also performed on the simulated data to compare its selection rates for one cluster (Table 2.1).

In the 200×4 data sets, all three permutation-based hypothesis tests are liberal tests with the rates of falsely rejecting the null hypothesis of no clustering structure higher than α. The cluster shuffling method with the F-statistic has the highest Type I error rate among the three methods (0.15), while the two versions of variable shuffling methods are close to $\alpha = 0.05$ (0.074 for AW statistic and 0.076 for F statistic). There is not a substantial difference in the rejection rates using either test statistic in the variable shuffling methods. Cross-validation-based criteria did not

Table 2.1: Percentage of unclusterable data sets that the methods consider as clusterable (out of 1000 simulated data sets). Overall $\alpha = 0.05$ for hypothesis tests.

	Rate of not choosing one cluster result		
Clustering method	200 x 4	200 x 8	300 x 4
Cluster shuffling	0.150	0.251	0.142
Permutation Variable Shuffle w/ AW	0.074	0.112	0.064
Permutation Variable Shuffle w/ F	0.076	0.111	0.065
minCV	1.000	1.000	1.000
CV1SE	0.994	0.596	1.000
CV2SE	0.419	0.029	0.804

perform well in this study. *MinCV* and *CV1SE* almost never correctly picked one cluster result, while *CV2SE* was wrong 42% of the time.

When the dimensions of the data increased from 4 to 8, both *CV1SE* and *CV2SE* methods improved (false rejection rate of *CV1SE* and *CV2SE* are down to 0.596 and 0.029, respectively). All permutation-based methods did worse (around 0.112 for both variable shuffling methods), while *minCV* was still not able to pick the one cluster result. Conversely, when the number of data points increased from 200 to 300 in 4 dimensions, the false rejection rates of the three permutation-based methods decrease while those of the three cross-validation-based methods increase.

When there is actually no cluster structure in the data set, regardless of the test statistic used in hypothesis tests, both variable shuffling permutation-based methods have a similar Type I error rate but neither is great. These methods are slightly liberal and they become more so when the data dimension increases. On the other hand, except for the *minCV* method, cross-validation-based methods did not perform well in the 4 dimensions case but improved in 8 dimensions, especially *CV2SE* with the error rate of only 0.029. When the number of observations increase, more information related to the lack of cluster structure becomes available and it may help lower the Type I error rate of the hypothesis tests.

2.5.2 Study 2: Accuracy in Choosing the Number of Clusters

When there is actual underlying cluster structure in the data, the ability to choose the "correct" number of clusters is important. In order to see how the different criteria perform in choosing number of clusters for different data generating mechanisms of data, we set up a simulation study including two scenarios with varying dimensions, partly inspired by simulations studies in Milligan and Cooper (1985) and Tibshirani et al. (2001).

1. *Scenario 1:* A random 4-cluster model in $Q = 4$ dimensions is generated from normal distributions. First, four values are sampled from a $\mathrm{N}(0, 5^2)$ distribution, then a multivariate normal distribution in four dimensions at means created and the correlation matrix of I_4 is used to generate the observations. Each cluster has 50 observations, making the total size of $n = 200$ for a simulated data set. In each data set, the distance between any pairs of clusters, defined as the Euclidean distance between the two closest observations from two clusters, similar to the "single linkage" in hierarchical clustering, is at least 2 units.

2. *Scenario 2:* By adding another variable with values generated from the standard normal distribution as a noise variable to the previous data set, we create another scenario for the criteria to choose the number of clusters. All the variables are standardized to avoid any variable dominating the distance between observations. Because of the additional noise variable, the four clusters

in these data sets are no longer clearly defined or well separated, creating a more challenging scenario for all the methods.

For each scenario, 500 data sets are generated. The methods of choosing the number of clusters mentioned in Sections 2.3 and 2.4 including average silhouette width (AW), CH's pseudo-F, three cross-validation-based criteria (minCV, CV1SE, and CV2SE), permutation hypothesis tests with pseudo-F and cluster shuffling, and permutation hypothesis test with variable shuffling (using AW and pseudo-F test statistics). Each method is applied to the cluster results created by monothetic clustering with the minimum and maximum numbers of clusters, K, from 1 and 10, respectively. The threshold α for permutation-based hypothesis testing methods is set as 0.05, and the p-value is calculated based on the null distribution from 1,000 permutations. The M-fold cross-validation was run with $M = 10$ to find the CV_K and SE of each clustering result K, then the three criteria (minCV, CV1SE, and CV2SE) are applied to pick the number of clusters. The AW and CH's pseudo-F were run using the **cluster** package (Maechler et al., 2018). The hypothesis test methods are implemented in our **monoClust** package with permutations done by the **vegan** (Oksanen et al., 2018) and **permute** (Simpson, 2016) packages.

The results of the simulation study with the number of cases that each method chose in each simulation scenario are summarized in Table 2.2 and Figures 2.6 and 2.7. In both scenarios, the minCV often favored the largest number of clusters considered. Ten clusters were picked 93.4% of the time in Scenario 1 and 96.8% in Scenario 2. This result suggests that the CV_K keeps decreasing when the number of clusters K increases. Unlike classification and regression trees where a response variable is used to assess the prediction errors, clustering re-uses the dissimilarity measures among the splitting variables to do that, so it does not seem to provide much penalty for over-fitting and hence is not suitable as a criterion for choosing the number of clusters.

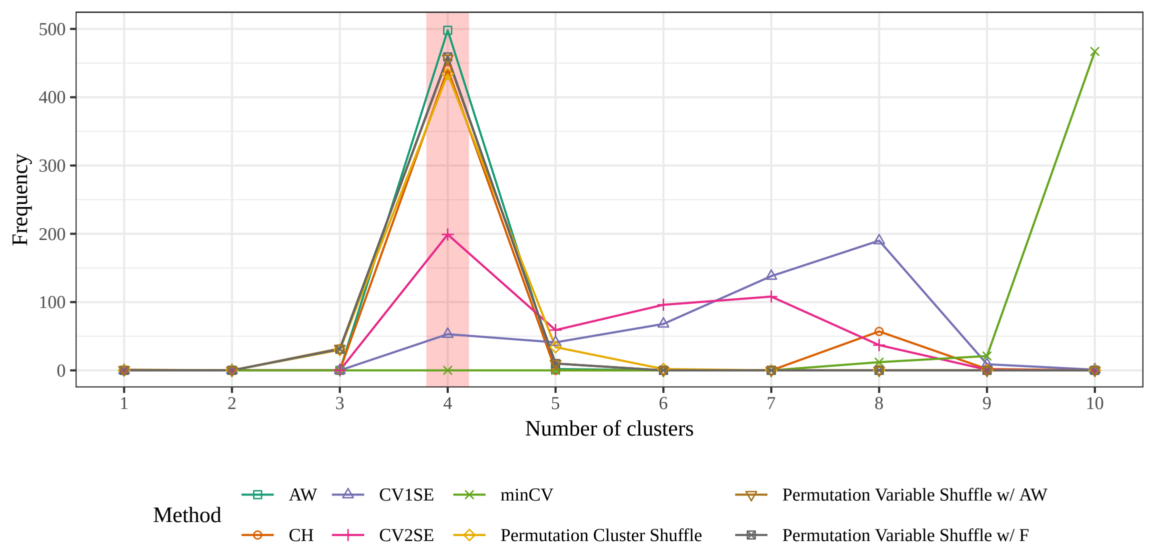


Figure 2.6: Simulation scenario 1 result. The frequency of choosing the number of clusters by various criteria for 500 simulated data with 4 variables, 4 clusters. The true number of clusters is highlighed with a red bar.

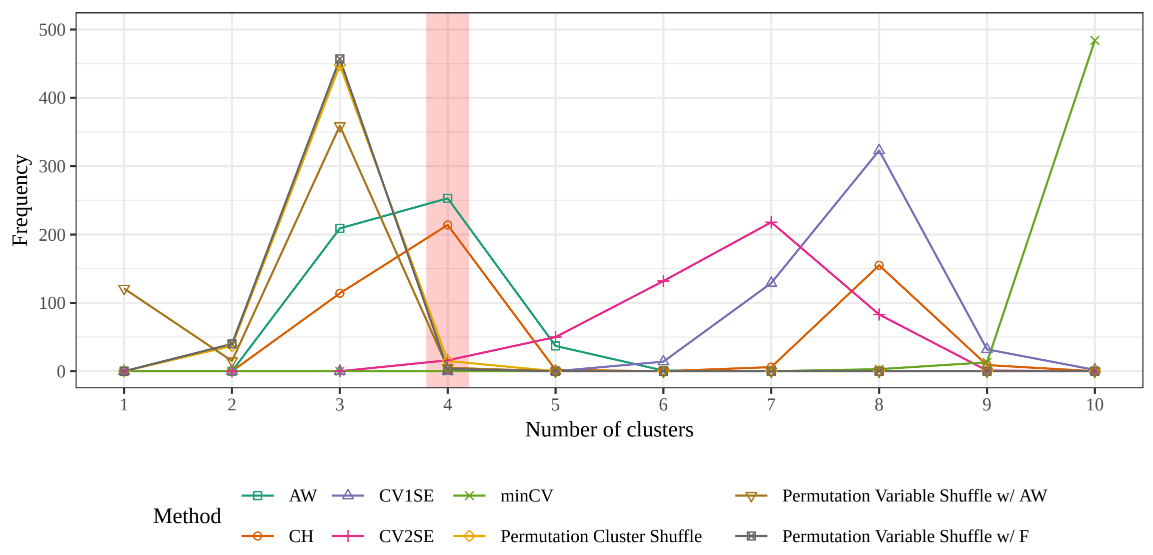


Figure 2.7: Simulation scenario 2 result. The frequency of choosing the number of clusters by various criteria for 500 simulated data with 4 clusters in 5 variables, in which one variable is normally distributed noise. The true number of clusters is highlighed with a red bar.

In Scenario 1, AW and CH's pseudo-F (CH) performed very well. They picked the correct number of clusters most the time (99.6% for AW and 88.2% for CH).

The CV-based criteria performed better when the standard error of the minimum error estimate is included to help detect the "elbow" decrease of the CV_K when K increases, with CV2SE performing better than CV1SE. The rate of correct identification of four clusters in Scenario 1 is 10.6% for CV1SE and 39.8% for CV2SE. Some mistakes in Scenario 1 might be due to the variability in dividing the data into 10-fold validation and training data sets. In a particular application, the CV selection process could be repeated and a consensus of results selected which would decrease the variability due to random variation in validation-split membership.

Permutation-based methods have a high rate of selecting the true number of data generating clusters in Scenario 1 and the variable shuffling methods performed slightly better than the cluster shuffling method (91.4% and 91.8% versus 86.8% for cluster shuffling). In this scenario, we again see that using either the AW or CH's F statistic in the permutation test does not create a substantial difference in the cluster results. They are very close to each other with CH's F only 0.4% points higher in the selecting four clusters than AW. The cluster shuffling method is also more liberal than variable shuffling methods, with the proportion of five cluster results at 6.8% compared to only 2% in the variable shuffling methods. This result is consistent with the simulation study on clusterability where the null hypothesis rejection rate of cluster shuffling is substantially higher than the other methods. This is due to the clusters created at each step by monothetic clustering already being the best possible in terms of inertia, and although the splitting variable is excluded in the test statistic calculations, any potential correlations between the splitting variable and other variables might have an effect in making the p-value smaller. The variable shuffling approach essentially creates a new data set in each permutation and repeats the optimization so it does not have this problem and hence is more precise in choosing the number of clusters.

In Scenario 2, it is obvious that the addition of the noise variable makes the underlying data generating clusters less separated, leading to more frequent identification of smaller numbers of clusters by all of the examined methods. The original AW and CH showed their resistance to noise and still pick the correct number of clusters most of the times (50.6% for AW and 42.8% for CH). Compared to the original AW and CH, the permutation-based methods are more sensitive to noise. In most cases, they supported choosing three clusters. It is also important to note that the variable shuffling method using AW as a statistic did not perform any splits for 24.2% of the cases, and is consistent with how the original AW performed with 41.8% three cluster results compared to only 22.8% three cluster results in the original CH. This suggests that there are situations where CH's F can outperform AW and that it is a better test statistic for the variable shuffling method.

2.5.3 Study 3: Hybrid Approach

In the previous sections, the original AW and CH's pseudo-F have been shown to perform very well in choosing the true number of clusters of simulated data sets. However, both of them cannot pick the one cluster solution. Meanwhile, the hypothesis testing approach (specifically, variable shuffling methods with CH's F as the test statistic) has a Type I error rate close to the nominal level but does not perform as well as AW or CH's pseudo-F when there is cluster structure in the data. Therefore, we suggest a hybrid technique where the variable shuffling hypothesis testing with CH's F statistic is used at the first split to test if the data set should have at least two clusters. If the test returns sufficient evidence for a split, the original CH's F is used to evaluate the number of clusters. This provides consistency of the statistic used (CH's F) but two ways that it is used.

In order to evaluate the improvement of the suggested hybrid approach, we set up another simulation study. This study has two groups of data sets with different numbers of true clusters. The first group includes 500 data sets with the size of 200×4, generated from a multivariate uniform distribution so that all observations belong to only one true cluster (similar to the study in Section 2.5.1). The second group includes another 500 data sets with the size of 200×4, generated by the same process as the study on the accuracy in Section 2.5 but with only two true clusters. The original CH's pseudo-F, the permutation-based variable shuffling with pseudo-F as the statistic, and the hybrid of these two are performed on these data sets. The summary of the cluster results chosen by the methods and the proportions of correct choice by each method is displayed in Table 2.3.

The hybrid method inherits the performance of the permutation test for the group of one true cluster data sets. This is the main benefit of the hybrid method over the original CH's pseudo-F which cannot choose one cluster. For the group of two true cluster data sets, the hybrid method inherits the benefits of the original CH's pseudo-F, with some loss due to the Type II errors of the permutation test at the first stage.

2.6 Remarks and Conclusions

Cluster analysis is a method to detect underlying structure when no information about that structure may be known. Therefore, choosing a reasonable number of clusters assists greatly in understanding and interpreting the characteristics of a data set. Monothetic clustering, with its ability to create binary splitting rules based on one variable at a time, has an advantage in interpretability over other clustering methods, but like all clustering methods, the number of clusters used impacts interpretations. Its similarity to CART also creates interest in exploring methods such as M-fold cross-validation and hypothesis tests at each binary split for selecting the number of clusters.

Several simulation studies were set up to evaluate various criteria. CH's pseudo-F and AW, the two methods that are commonly used and have been shown to be effective in some data structures, also performed very well in monothetic clustering. When the cluster structure is well-defined and is normally distributed around the cluster means, these two methods picked the correct number of clusters most of the time, with AW slightly better than CH's F. Permutation-based hypothesis tests at each node, especially the pair of variable shuffling approaches, did not perform as well as AW and CH's F in picking the correct results in the simulation study with four true cluster data sets. However, they really shined when no cluster structure exists in the data set. The hypothesis tests are able to correctly choose the one cluster result in the simulation study with the false rejection rate only a bit higher than the nominal significance level α.

Based on these simulation results, we suggested a hybrid approach to sequentially apply the permutation test at the first split and then apply the original CH's F in further splits. This hybrid method shows its potential in improving the ability to choose the correct number of clusters when the data sets are on the edge of having one or two clusters. This approach could even have a higher rate of picking a reasonable number of structures when the number of variables is high (making the permutation based test less liberal at the first split) and the cluster structure is well-defined and/or well-separated (making CH's F criterion more effective).

Cross-validation-based approaches inspired from CART did not work effectively in our simulation study with monothetic clustering. The fact that CV_K keeps decreasing when the number of clusters, K, increases (at least until 10 clusters) makes it difficult to detect the "elbow" position in the CV plot. The cluster shuffling approach seems to be overly liberal and tends to choose a higher number of clusters than correct even with the Bonferroni's corrected p-values. Shuffling observations between two candidate clusters is not sufficient to break the cluster separation created by the split for the best inertia of monothetic clustering and thus does not create a good null distribution.

There is potential to further explore the problem of choosing the number of clusters in monothetic clustering. Although the suggested permutation-based variable shuffling approach is competitive with the commonly used original AW and CH's F, the permutation null hypothesis could be better formulated so the calculated p-value of the hypothesis tests can be interpreted in terms of the clustering problems. Because there is no “best” criterion for all clustering problems, other simulation studies on data sets with different cluster structures (such as elongated-shaped clusters, or clusters defined by splitting variables) to examine scenarios where the proposed criteria may perform better than the original AW and CH's F. All of the suggested criteria in this chapter have been implemented in the **monoClust** package in R (see Chapter 5 for more details).

Table 2.2: The proportions of cluster results chosen by methods of choosing number of clusters.

	Number of clusters chosen									
Method	1	2	3	4	5	6	7	8	9	10
Scenario 1										
AW	0.00	0.00	0.00	**1.00**	0.00	0.00	0.00	0.00	0.00	0.00
CH	0.00	0.00	0.00	**0.88**	0.00	0.00	0.00	0.11	0.00	0.00
CV1SE	0.00	0.00	0.00	**0.11**	0.08	0.14	0.28	0.38	0.02	0.00
CV2SE	0.00	0.00	0.00	**0.40**	0.12	0.19	0.22	0.07	0.00	0.00
minCV	0.00	0.00	0.00	**0.00**	0.00	0.00	0.00	0.02	0.04	0.93
PCS[a]	0.00	0.00	0.06	**0.87**	0.07	0.00	0.00	0.00	0.00	0.00
PVS-AW[b]	0.00	0.00	0.06	**0.91**	0.02	0.00	0.00	0.00	0.00	0.00
PVS-F[c]	0.00	0.00	0.06	**0.92**	0.02	0.00	0.00	0.00	0.00	0.00
Scenario 2										
AW	0.00	0.00	0.42	**0.51**	0.07	0.00	0.00	0.00	0.00	0.00
CH	0.00	0.00	0.23	**0.43**	0.00	0.00	0.01	0.31	0.02	0.00
CV1SE	0.00	0.00	0.00	**0.00**	0.00	0.03	0.26	0.65	0.06	0.00
CV2SE	0.00	0.00	0.00	**0.03**	0.10	0.26	0.44	0.17	0.00	0.00
minCV	0.00	0.00	0.00	**0.00**	0.00	0.00	0.00	0.01	0.03	0.97
PCS[a]	0.00	0.07	0.90	**0.03**	0.00	0.00	0.00	0.00	0.00	0.00
PVS-AW[b]	0.24	0.03	0.72	**0.01**	0.00	0.00	0.00	0.00	0.00	0.00
PVS-F[c]	0.00	0.08	0.91	**0.01**	0.00	0.00	0.00	0.00	0.00	0.00

[a] Permutation Cluster Shuffle

[b] Permutation Variable Shuffle w/ AW

[c] Permutation Variable Shuffle w/ F

Table 2.3: The proportions of correctly picking true number of clusters for original CH's F, variable shuffling with F as test statistic, and the hybrid of the two.

	One true cluster			Two true clusters			
Clustering method	**1**	> 1	Rate	1	**2**	> 2	Rate
CH's F	**0**	500	0.00	0	**500**	0	1.00
PVS-F[a]	**462**	38	0.92	25	**455**	20	0.91
Hybrid	**462**	38	0.92	25	**475**	0	0.95

[a] Permutation Variable Shuffle w/ F

CHAPTER THREE

VISUALIZATION AND MONOTHETIC CLUSTERING DATA WITH CIRCULAR VARIABLES

Contribution of Authors and Co-Authors

Author: Tan V. Tran

Contributions: Responsible for the majority of the writing.

Co-Author: Mark C. Greenwood

Contributions: Provided guidance and feedback on statistical analysis and drafts of the manuscript.

Co-Author: John C. Priscu

Contributions: Provided guidance in interpretation of the results.

Co-Author: Marie Šabacká

Contributions: Collected and described data.

Manuscript Information

Tan V. Tran, Mark C. Greenwood, John C. Priscu and Marie Šabacká

Journal of Environmental Statistics

Status of Manuscript:

__X__ Prepared for submission to a peer-reviewed journal

_____ Officially submitted to a peer-reviewed journal

_____ Accepted by a peer-reviewed journal

_____ Published in a peer-reviewed journal

Published by the UCLA Department of Statistics

Abstract: Monothetic clustering is a clustering technique that has advantages in interpretability and prediction compared to other clustering techniques. It is based on recursive bipartitions of a data set by choosing splitting rules on the variables, one at a time. Circular variables, the type of variables whose values are defined within a range where the upper bound coincides with the lower bound (e.g., directional variables, time of day, etc.), must be treated differently from conventional quantitative variables. In this paper, we suggest a clustering procedure for a data set that contains circular variables, starting from visualization, to calculating an appropriate distance matrix, to proposing a clock's hour and minute hand-type partitioning that can be used in monothetic clustering. We apply the proposed method to a data set on air particle count measured in Antarctica to illustrate the utility of the method. Using data measured every 15 minutes, the particle counts, wind speed, and wind direction are used to create meaningful rules to partition the multivariate data set. By doing that, we can identify groups of wind conditions with shared patterns in sediment transport in the Taylor Valley in Antarctica.

3.1 Introduction

In many applications, a variable can be measured in angles, indicating the directions of an object or event. Examples could be the time of day, aspect of the slope in mountainous terrain, direction of motion, or, here, wind directions in a study interested in counting the number of particles carried in the air in Antarctica. Such variables are referred to as *circular variables* and are measured either in degrees or radians relative to a pre-chosen 0 degree position and direction of rotation. For circular variables, the highest and lowest values are the same value, but this is not true for typical quantitative variables.

The analysis of data sets including circular variables requires different sets of statistical methods from conventional “linear” quantitative variables because of the unique characteristics of circular variables. There are books dedicated to this topic (for example, Fisher, 1993; Jammalamadaka and SenGupta, 2001; Pewsey et al., 2013) that develop parametric models and analytic tools for circular variables. They address statistical techniques focused on univariate visualization and modeling with circular variables but do not mention the topic of multivariate data analysis involving circular variables, such as the multivariate visualization and clustering that are discussed here.

Cluster analysis (or clustering) attempts to group observations into groups, or clusters, so that the multivariate observations within a cluster are similar to each other but different from those in other clusters (Everitt and Hothorn, 2011, Chapter 6). Let y_{iq} be the i^{th} observation ($i = 1, \ldots, n$, the number of observations or sample size) on variable q ($q = 1, \ldots, Q$, the number of response variables) which can be conventional quantitative, categorical, or circular variables in a data set. Clustering algorithms then attempt to partition the n observations into mutually exclusive clusters $C_1, C_2, \ldots, C_K$ in Ω where K is the number of clusters and Ω is the entire set of observations, so that the observations within a cluster are “close” to each other and “far away” from those in other clusters.

Commonly used clustering methods such as k-means or hierarchical clustering with Ward’s method belong to a group of methods called polythetic clustering (MacNaughton-Smith et al., 1964) that use combined information of variables to partition data. Clusters created are similar “on average” but may share no common characteristics. In contrast, monothetic cluster analysis (Chavent, 1998; Piccarreta and Billari, 2007; Sneath and Sokal, 1973) is a class of clustering algorithms that provides clusters with shared characteristics using a hierarchical, recursive partitioning of multivariate responses based on binary decision rules that are built from individual response variables. By construction, members of the same clusters will share common characteristics such as the same category of a categorical response or in the same interval of a quantitative variable.

Inspired by classification and regression trees (Breiman et al., 1984), the monothetic clustering algorithm of Chavent (1998) searches for splits from each response variable that provide the best split of the multivariate responses in terms of a global criterion called inertia. It then recursively applies the same algorithm to each sub-partition, recording splitting rules to define the tree. The result of the algorithm is a set of hierarchical binary rules for determining cluster membership. Therefore the resulting hierarchy can be read and displayed as a decision tree. The decision tree defines the minimum set of shared characteristics.

While Chavent (1998) used classic data sets, *ruspini* (Ruspini, 1970) and *iris* (Fisher, 1936), to demonstrate monothetic clustering algorithm, Chavent et al. (2007) compared the monothetic clustering to k-means and Ward's method by both simulation and six applied data sets (three data sets with numerical and three data sets with categorical variables) from the UCI Machine Learning repository (Dua and Graff, 2019). They found that monothetic clustering performed better than k-means and Ward's methods when the number of clusters is small. Also, for numerical data sets, the larger the data set is, the better result mononethic clustering has. Piccarreta and Billari (2007) have also used the same monothetic divisive clustering idea to group women based on their life course states (marriage, child-bearing, and having a job), creating groups of life course trajectories based on the combination of work and family orientation. By doing that, they are able to interpret the resulting clusters using the participants' life course states due to the rules defining those clusters.

To do a cluster analysis, one has to decide how to measure the distance or dissimilarity between data points. While there are many distance or dissimilarity metrics for conventional linear, categorical, and mixed variables (Everitt et al., 2011; James et al., 2013), Euclidean distance is the most common choice:

$$d_{euc}(\mathbf{y_i}, \mathbf{y_j}) = \sqrt{\sum_q (y_{iq} - y_{jq})^2}. \tag{3.1}$$

It satisfies all distance metric properties, including:

1. Non-negativity: $d(y_1, y_2) \geq 0$,
2. Identity of indiscernibles: $d(y_1, y_2) = 0 \Leftrightarrow y_1 = y_2$,
3. Symmetry: $d(y_1, y_2) = d(y_2, y_1)$, and
4. Triangle inequality: $d(y_1, y_3) \leq d(y_1, y_2) + d(y_2, y_3)$.

If a distance measure satisfies these properties, it is considered a *distance metric* (Arkhangel'skii and Pontryagin, 1990).

For our application, it is important that the multivariate distance needs to work for data sets with both conventional and circular variables. Lund (1999), while examining least squares regression for a model involving circular response and/or predictors, suggested a distance measure between two circular observations, y_1 and y_2, by applying a trigonometric cosine transformation of the difference in their angles (in radians or degrees) around the circle as

$$d_{lund}(y_1, y_2) = \frac{1}{2}\left(1 - \cos(y_1 - y_2)\right). \tag{3.2}$$

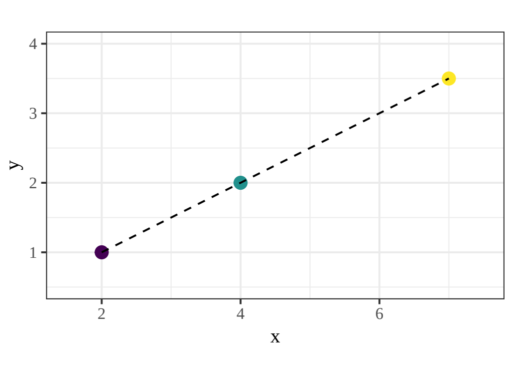
This dissimilarity measure satisfies all of the distance metric properties, making it a true distance metric. In a related paper, Lund (2002) proposed that when a circular variable q is used as a response variable in a regression tree problem setting, the impurity measure (target function) of a tree node is the mean distance using similar dissimilarity from all observations y_{ij} to its circular mean direction of observations in that node, $\overline{y}^*_{\cdot j}$,

$$R(t) = \frac{1}{n_t} \sum \left[1 - \cos(y_{ij} - \overline{y}^*_{\cdot j})\right].$$

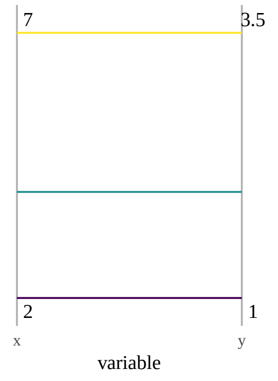
An alternative metric that maintains all four properties of a distance metric is proposed below.

The Parallel coordinates plot (PCP) is a means to display multi-dimensional data where Q variables are represented by Q lines, placed equidistant and perpendicular to the traditional x-axis (hence the name parallel coordinates). A multivariate data point y_i $(1 \times Q)$ that has the coordinates $(y_{i1}, y_{i2}, \ldots, y_{iQ})$, which is a point in Euclidean Q-dimensional space, is a polygonal line whose q-th vertex is at the y_{iq} value on the q-axis for $q = 1, 2, \ldots, Q$ (Inselberg and Dimsdale, 1987). Because points in Euclidean space can be displayed in PCPs as line segments, the linear dependencies between variables can be detected in PCPs. Mostly parallel lines in PCP are evidence for a positive linear relationship between neighboring variables, mostly intersecting or crossing lines are evidence for a negative linear relationship, and other patterns may discerned by tracking combinations of results across the Q variables (see Figure 3.1 for a visual explanation). Another useful characteristic of PCPs as a companion to cluster analysis is their ability to display and explore the identified sub-groups. Finally, the convergence of lines to discrete values of a (discrete or categorical) variable is visual evidence of sub-groups in the data set with common characteristics (Härdle and Simar, 2015, Chapter 1).

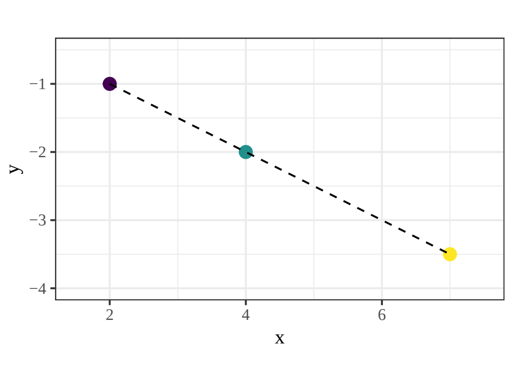
We add options to work with circular data as well as a method for visualizing cluster solutions with circular variables and using monothetic clustering with circular variable(s) in the following sections. These methods are described and illustrated with an application to a data set of particle counts in the air in Antarctica originally



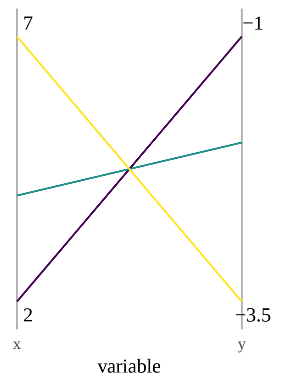
(a) Perfect positive linear relationship.



(b) Parallel lines in PCP.



(c) Perfect negative linear relationship.



(d) Crossing lines in PCP.

Figure 3.1: The points in Euclidean coordinates (left) and their representations in Parallel coordinates plots (right). The variables in PCPs are scaled to have the same length with the minimum and maximum values shown.

analyzed in Šabacká et al. (2012).

3.2 Visualization of Circular Variables

Circular variables are usually displayed on circles, either with the data points plotted on the circle, or by displaying the angles in the circle. Many popular plots in exploratory data analysis such as the dot-plot, histogram, stem and leaf, and density plot have circular versions. A unique plot for circular data that is often used to examine the distribution of a circular variable is the *rose diagram*, which was used as early as 1858 to depict the number of deaths caused by contagious diseases, wounds, and other reasons for the British Army in hospitals, by months of the years 1855-1856 (Fisher, 1993). The author used a rose diagram with different colors for the causes to visualize the importance of sanitation in hospitals across the year (the months were treated as circular). The rose diagram is very similar to a circular version of a histogram but the bars are narrower toward the center of the circle to form wedges that correspond to the frequency of responses in that bin of angles, thus creating an overall symmetric shape for the plot. Figure 3.2 is an example of a combination of a rose diagram, a circular dot-plot, and a circular density plot on a circle for wind direction data measured in the Bonney Reigel location in the McMurdo Dry Valleys in the Antarctica (more details in Section 3.4.1). In these plots, 0 and 360 degrees correspond to the winds coming from the north and clockwise rotation relates to increasing degrees (from the east is 90 degrees). One can see that the rose diagram has similar information to the circular dot-plot and density plot, showing that the recorded wind directions mostly came from the angles close to 280 and 100 degrees, which generally corresponds to the orientation of the valley in which the sensor was placed. There are also some winds that come from 200 degrees flowing from the peak to the side of the mountain (see the map in Figure 3.7 for context of the site).

When clustering a data set that has at least one circular variable in it, visualizing the cluster results to detect the underlying characteristics of the clusters is important. A rose diagram with wedges (or stacked wedges if observations from multiple clusters happen to be in the same wedge) colored by cluster membership can visualize the cluster solution on the circular variable but fails to show the results on other variables and is difficult to envision a linked multivariate graph as we will demonstrate below. Scatterplots with circular variables end up with close observations far apart for observations near $0/360^{o}$, so are of limited ability with circular data.

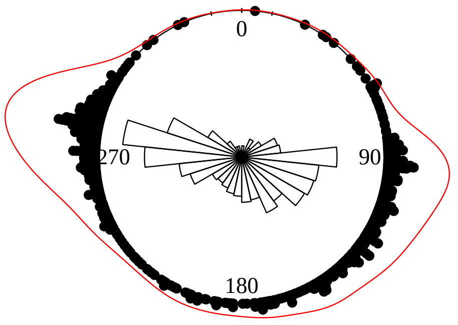


Figure 3.2: A circular display of the distribution of the wind directions measured at the Bonney Riegel location in the McMurdo Dry Valleys in the Antarctica from July 7 to 14, 2008. They were recorded every 4 seconds and were averaged at 15 minute intervals. The plot includes rose diagram (bars), dot-plot (dots around circle), and non-parametric density plot (red line outside or on circle).

Because PCPs are useful in describing clusters in a data set with numerous variables, by displaying each observation uniquely and with variables on separate axes, it has the potential to be a good tool to show the circular variables along with other conventional variables in a data set. There are at least three ways to consider integrating a circular variable into a PCP. First, the circular variable can be displayed as a conventional linear variable with the values range from 0 to 360 degrees. This approach has a serious flaw in that the distances between values on the circle are not maintained correctly, especially between values around 0 and 360 degrees, which in fact, coincide.

The second way to plot a circular variable q in a PCP is to exploit the transformation of polar coordinates to rectangular coordinates. Let y_{iq} be a value of variable q for observation i, measured in an angle unit (degrees or radians).

Its polar coordinate on a unit circle is (1, y_{iq}) and its corresponding rectangular coordinate is $(1 \times \sin(y_{iq}), 1 \times \cos(y_{iq}))$. The two variables $\sin(y_{iq})$ and $\cos(y_{iq})$ are no longer circular variables (there is neither coincidence between 0 degrees (0 radians) and 360 degrees (2π radians) nor the need to choose a "positive" direction for the variables' values), but they are dependent on each other by the unit circle constraint, $\sin^2(y_{iq})+\cos^2(y_{iq}) = 1$. Consequently, a PCP with $Q+1$ vertical axes can be drawn to visualize multivariate data with a circular variable as in Figure 3.3 that shows the PCP of the monothetic clustering of the Antarctica data set discussed previously. The wind direction variable is transformed into sin(*WDIR*) and cos(*WDIR*), shown together with two conventional variables, whether the wind has particles or not (*has.sensit* as a binary variable) and the wind speed (*WS* as a quantitative variable). We can see that the lines have dependencies at their extremes between sin(*WDIR*) and cos(*WDIR*), creating hyperbolae at the top and the bottom of the area between the two axes. The constraint of lying on the unit circle for these two variables combined compromises the easy interpretation and use of these variables in a PCP. A PCP display of increasing values of variables in the bottom-up manner from the smallest value (-1) to the largest value (1) negatively affects the presentation of angular values in the sine and cosine of the angular variables. A "full circle" of sine and cosine passes the values from -1 to 1 twice, making the lines frequently overlapped on sine and cosine axes. It is difficult to distinguish different pairs in the densely intertwined lines connecting between sine and cosine. It is also difficult to extract the original angles from the plot of their sine/cosine pairs.

The third option builds on the idea in Will (2016) who proposed an adaption of PCP to include circular variables by depicting them with a pseudo three-dimensional circular aspect. In particular, circular variables are drawn as ellipses on their own axes within the PCP, as shown in Figure 3.4. Ellipses give readers a 3-D feel of

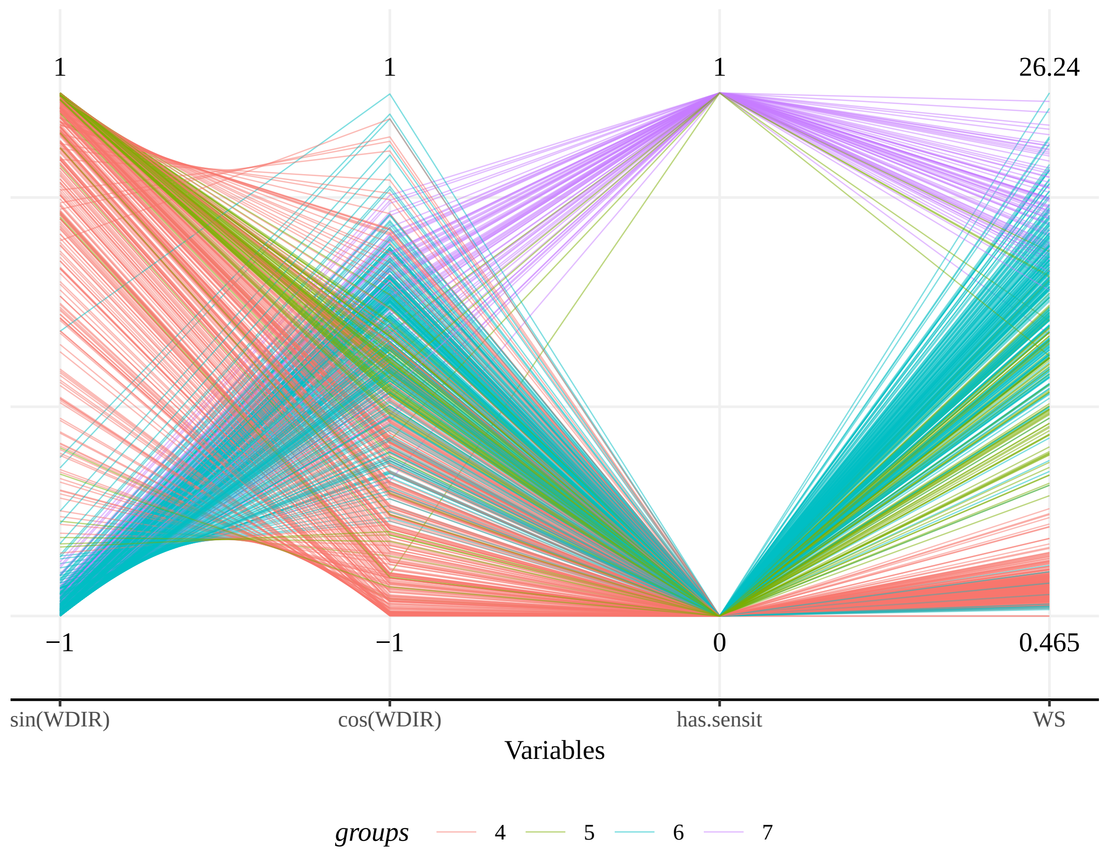


Figure 3.3: PCP with the circular variable ($WDIR$) replaced by its trigonometric transformations using sine and cosine for Antarctica data set from July 7 to 14, 2008. Lines are drawn in colors corresponding to the clusters found by monothetic clustering with four cluster solution.

circular variables and all points on a circle can be plotted on an ellipse correctly. However, the lines on the left and right sides of the ellipse can still overlap, especially when they connect to similar values on the adjacent axes (variables). We expand this idea by implementing the ability to rotate for the ellipses in our software package. Using ellipses with the ability to rotate can help distinguish observations, potential clusters, and the geographical directions (when needed). A careful choice of rotation and order of variables neighboring the circular variable on PCPs can help alleviate this overlapping.

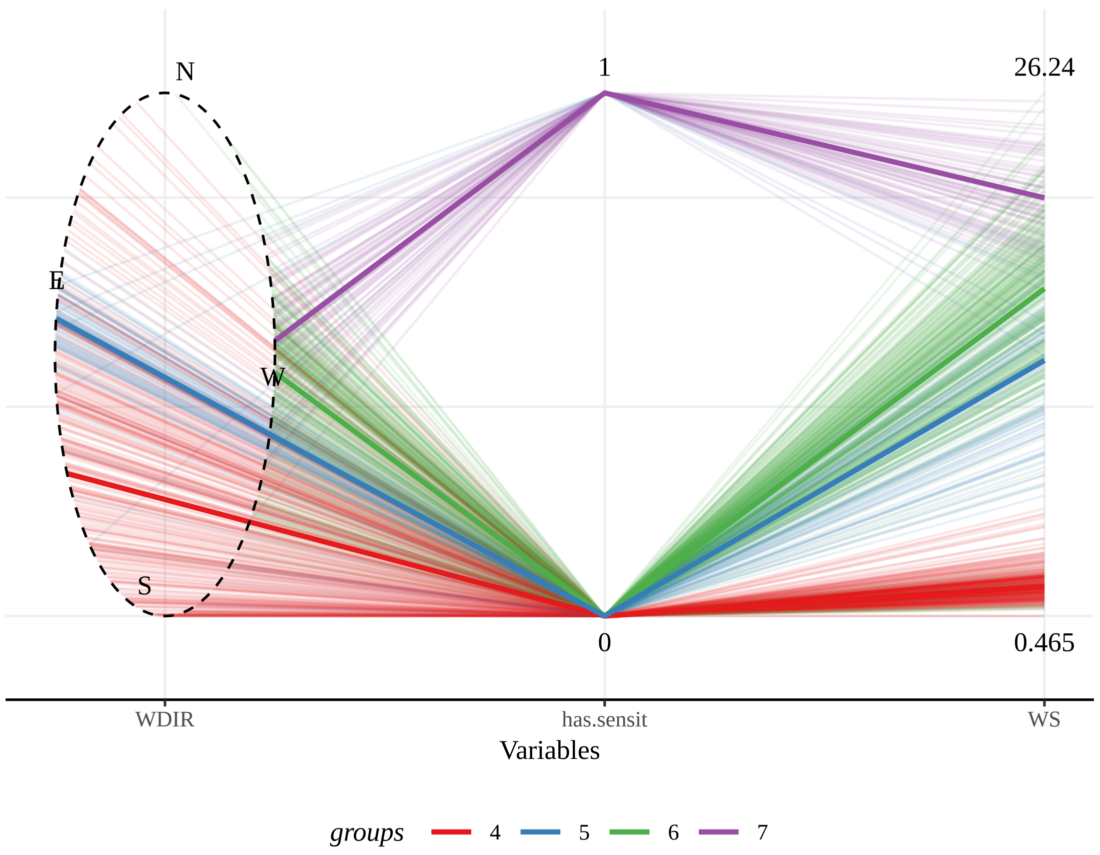


Figure 3.4: PCP with circular variable ($WDIR$) is depicted as an ellipse for particle count for Antarctica data set from July 7 to 14, 2008. The geographical direction is noted and the ellipse is rotated to facilitate understanding of clusters. Clusters in the plot are based on monothetic cluster analysis discussed later.

3.3 Monothetic Clustering with Circular Variables

In monothetic clustering, the within cluster inertia, $I(C_k)$, of a cluster C_k is a criterion used for deciding on partitions of the data by measuring the total variability around the cluster centroid. In the case of Euclidean distance, the inertia for cluster C_k would be

$$I(C_k) = \sum_{i \in \mathrm{C_k}} \sum_{q=1}^{Q} (y_{iq} - \overline{y}_{\cdot q})^2, \tag{3.3}$$

where $\overline{y}_{\cdot q}$ is the mean of all observations y_{iq} in variable q. Let $s(C_k)$ be a binary split dividing a cluster C_k into two clusters C_{kL} and C_{kR}. The decrease in inertia is

$$\Delta(s, C_k) = I(C_k) - I(C_{kL}) - I(C_{kR}). \tag{3.4}$$

The objective of the algorithm when applying a splitting rule s to cluster C_k is to maximize the difference in inertia between C_k and the new sub-partition C_{kL} and C_{kR},

$$s^*(C_k) = \arg\max_{s} \Delta(s, C_k), \tag{3.5}$$

which is then recursively applied to each sub-partition.

James et al. (2013) and others have shown that the sum of Euclidean distances for all observations within a cluster and variation around the means within a cluster are proportional, so the within cluster inertia can be equivalently calculated from the dissimilarity matrix as

$$I(C_k) = \frac{1}{n_k} \sum_{(i,j) \in C_k, i>j} d^2_{euc}(\mathbf{y_i}, \mathbf{y_j}), \tag{3.6}$$

where n_k is the cardinality (size) of C_k. This result is used to justify the application of many distance-based methods to non-Euclidean dissimilarities (Anderson, 2001) used to calculate $d(\mathbf{y_i}, \mathbf{y_j})$'s.

One (dis)similarity measure for mixed data types is Gower's distance (Everitt et al., 2011, Chapter 3). Gower (1971) proposed a similarity measure between observations from various types of variables: quantitative, categorical, and binary. It can be used in Equation 3.6 when converted to a dissimilarity (by subtracting the similarity measure from 1). The general Gower's dissimilarity between two observations in a data set with Q variables is

$$d_{gow}(\mathbf{y_i}, \mathbf{y_j}) = \frac{\sum_{q=1}^{Q} w(y_{iq}, y_{jq}) d_{gow}(y_{iq}, y_{jq})}{\sum_{q=1}^{Q} w(y_{iq}, y_{jq})},$$

where $d_{gow}(y_{iq}, y_{jq})$ is the Gower's dissimilarity between observations y_{iq} and y_{jq} with regard to the variable q and $w(y_{iq}, y_{jq})$ is the weight coefficient. Gower's distance can accommodate some missing values, with $w(y_{iq}, y_{jq})$ set to 0 if the value is missing for either or both of the observations and $w(y_{iq}, y_{jq})$ is set to 1 if both are available. This drops the comparison between two observations on that variable and re-weights the resulting dissimilarity. Gower's distance formula provides distances, $d(y_{iq}, y_{jq})$ and $d_{gow}(\mathbf{y_i}, \mathbf{y_j})$, between 0 and 1.

Gower's distance can incorporate categorical variables where $d(y_{iq}, y_{jq})$ is 0 if the two observations belong to the same category of q and 1 otherwise. If q is a linear quantitative variable,

$$d(y_{iq}, y_{jq}) = \frac{|y_{iq} - y_{jq}|}{\max_{i,j} |y_{iq} - y_{jq}|}$$

is a scaled Manhattan (city block) distance, which is 0 when y_{iq} and y_{jq} are the same and 1 for the maximum difference.

We suggest the dissimilarity measure for a circular variable,

$$d(y_{iq}, y_{jq}) = \frac{180 - |180 - |y_{iq} - y_{jq}||}{180}.$$

The formula was modified from Jammalamadaka and SenGupta (2001) to provide distances based on the shortest distance around the circle, in relative degrees (or radians if suitably modified from 180 to π in the above formula), bounded in the [0, 1] interval. We propose its use over Lund's suggested distance because a) it also satisfies all of the properties of a distance metric, b) it retains some properties of

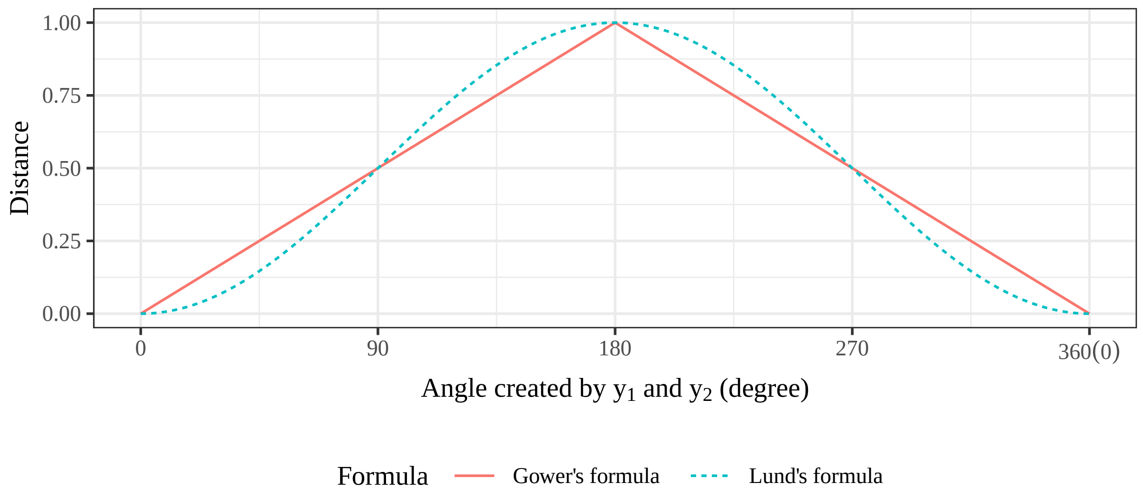


Figure 3.5: The Lund's and Gower's inspired distances between two values (in degrees) of a circular variable.

Euclidean distances (Pavoine et al., 2009) and can be used in Gower's dissimilarity with other variables, and c) if two pairs of observations have equal distance around the circle, they should be considered equally "far apart" both visually and conceptually. This is useful for understanding and interpreting the results of monothetic clustering where splits are made at raw values of variables. The visual comparisons between the preferred absolute distance and Lund's distance for circular variables are plotted in Figure 3.5.

Lund (2002) also discussed circular variables in the context of a regression tree where they were used as response as well as explanatory variables. He suggested that when a circular predictor is considered for a split, it is different from the conventional linear variable in that two values α_1 and α_2 are necessary to form two non-overlapping arcs. Figure 3.6 explains how two values are needed to make the first binary split but only one value to make the subsequent one. Moreover, all pairs of (α_1, α_2) need to be examined to determine which pair optimizes the target function. We adopt this idea

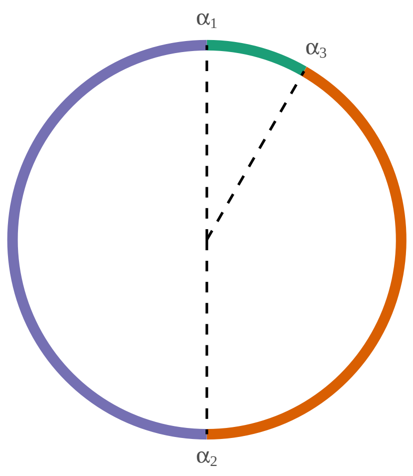


Figure 3.6: An example of binary splits of a circular variable. Two points (at α_1 and α_2) are necessary to split a circle into two non-overlapping arcs. The subsequent binary split needs only one more point (α_3) to split observations between (clockwise) α_1 and α_2.

into monothetic clustering and let the algorithm search for all possible pairs of arcs by first keeping the first cut fixed and rotating the second cut, similar to the idea of the hour and minute hands of a clock. If that circular variable is considered again further down the tree, its arc can be treated as a linear variable and the best split is determined according to that.

3.4 Application of Monothetic Clustering to Antarctic Particle Counts Data

3.4.1 Antarctic Particle Counts Data

The motivating data set for these methods is a part of a study on microorganisms carried in strong föhn winds at the Taylor Valley, an ice free area in the Antarctic continent (Michaud et al., 2012; Šabacká et al., 2012). Föhn winds are defined to be dry warm winds that flow from the top to the bottom of mountain ranges during the “flow over” effect (Elvidge et al., 2016). Šabacká et al. (2012) were interested in exploring relationships between wind behavior and sediment transport. To study this, sensit H11B acoustic wind mass erosion particle counters were deployed in five locations within the Taylor Valley: Bonney Riegel (BR), Bonney (BON), Hoare (HOR), FRX (Fryxell), and Lake Fryxell (F6) (map of these locations is in Figure 3.7). The BR location is in the Upper Valley area which was found to have higher particle flux than mid and lower valley areas so it would have more non-zero particle observations, making it more interesting for our purposes here. We analyze data from a particle counter at BR that was mounted on a meteorological station tripod at around 20 cm above the ground surface. The detector sampled for 1 minute every 15 minutes during the study to save power. Wind direction and wind speed data were obtained from the meteorological station. Wind direction was recorded every 30 seconds and wind speeds every 4 seconds at 1.15 meters above the ground surface. The recorded wind directions and speeds were averaged at 15 minute intervals. For wind direction, as discussed previously, winds from the north are defined as 0/360° and from the east as 90°. Although the authors examined the relationship between pairs of recorded variables, no analysis was done that explored their multivariate relationship and patterns. In this study, we apply cluster analysis to this data set to explore the common patterns in BR and try to detect groups of observations related to föhn winds along the valley.

Although the amount of collected data is large, only 2.39% of all particle counter measurements were non-zero. Therefore, for the purpose of this study, we only examine the data during August 4–8, 2007 and July 7–14, 2008 at the BR site to illustrate the methods. Those are the time spans when the sensors somewhat frequently recorded non-zero particle counts in the winter time (13.71% for the 2007 period and 10.85% for the 2008 period). Moreover, due to strong periodic föhn winds in the area, the highest particle flux can sometimes surge to a very high value ($\approx$7000

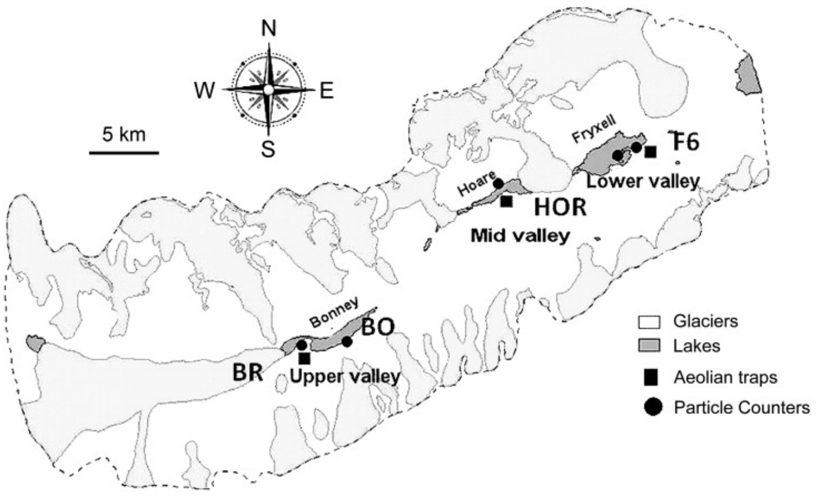
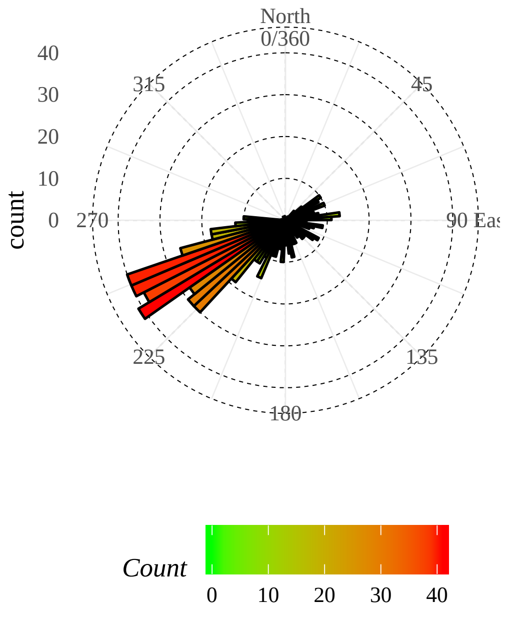


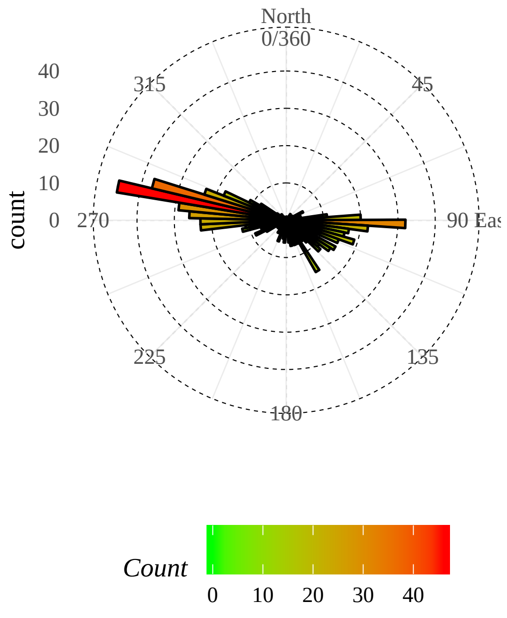
Figure 3.7: Map of the sensors in Taylor Valley, Antarctica. The site of interest, Bonney Riegel, is the circle with BR abbreviation.

particles per minute) although counts are usually much less than 500 particles per minute. This makes the particle count variable extremely right-skewed in this time frame. It strongly affected the distance measures between observations in any cluster analysis when the counts were considered in the Gower's distance. To overcome this problem, we transformed this variable into a binary one with existence/non-existence of particles for each measurement. This shifts the interpretation to focus on detection of any feature related to the existence (or lack there of) of aeolian flux as opposed to amount of particles being transported.

The wind direction variable is displayed in rose diagrams of the frequency of recorded wind directions in the two time spans in Figure 3.8. In both of them, the wind is mainly flowing in the East-West direction, in the lee of the Taylor Valley (Figure 3.7) with the West to East direction (from the inland to the sea) happening more frequently than the opposite due to the föhn wind direction in winter. A small difference in the typical directions of the wind between the two time periods can be attributed to the position of the BR location in the valley and a switch in wind



(a) August 4–8, 2007



(b) July 7–14, 2008

Figure 3.8: Rose diagrams showing the distribution of particle existence recorded by wind directions in two time periods at BR location in Taylor Valley. North to South winds are at 0/360° and East to West winds are at 90°. The angles indicate the directions "from" which the winds flowed.

patterns to have some winds that go across the valley connecting ice free areas on two sides of a glacier.

3.4.2 Results

The monothetic clustering algorithm was applied to two previously described subsets of the data in BR using wind speed (*WS*), wind direction (*WDIR*), and particle existence (*has.sensit*) variables. Because Gower's distance put equal weights on all the variables, we did not need to scale the variables before applying the algorithm. A permutation-based test for the presence of a cluster structure was done (see Chapter 2 for more details on this procedure). After getting strong evidence that there is some cluster structure in both of the 2007 and 2008 data sets (p-value < 0.001), we applied monothetic clustering algorithm to decide the clusters in the data. To keep the interpretation simple as a demonstration example, the number of clusters is chosen to be 4 and 6 for each data set and their results are compared.

The summary of the splitting rules for the 2007 data with four clusters is in Table 3.1 and graphically in Figure 3.10b. All of the observations having low wind speed ($WS < 4.85$ m/s) belong to a cluster (Cluster ID 2) which includes $n = 277$ 15-minute intervals. The data points having stronger wind speed ($WS \geq 4.85$ m/s) are further split into 2 smaller clusters depending on whether the particles were recorded (Cluster ID 7) or not. Another partition was made separating the observations with very strong wind ($WS \geq 16.45$ m/s, Cluster ID 13) and the rest (Cluster ID 12) given that they all did not contain particles (or the particles were too small to be recorded by the sensit sensor).

When the additional two splits are further made to create a six cluster solution, the wind direction variable is chosen by the algorithm to make both splits (Table 3.2). The splitting rule at the wind direction essentially split the observations into Easterly or Westerly winds. The cut occurs into a low wind speed cluster (Cluster ID 2 into Cluster IDs 4 and 5) and a high wind speed with particles recorded group (Cluster ID 7 into Cluster IDs 14 and 15).

The reason why wind direction was not introduced earlier in the monothetic clustering process can be explained by looking at the PCPs with the cluster medoids highlighted in Figures 3.10b and 3.10d. Despite wind direction not showing up in the splitting tree for the four cluster solution, the cluster members are mostly close together in the *WDIR* variable. This indicates a possible strong relationship between the wind speed (the variable at the first split) and wind direction where the wind tended to be much stronger in one direction compared to the other. Moreover, we

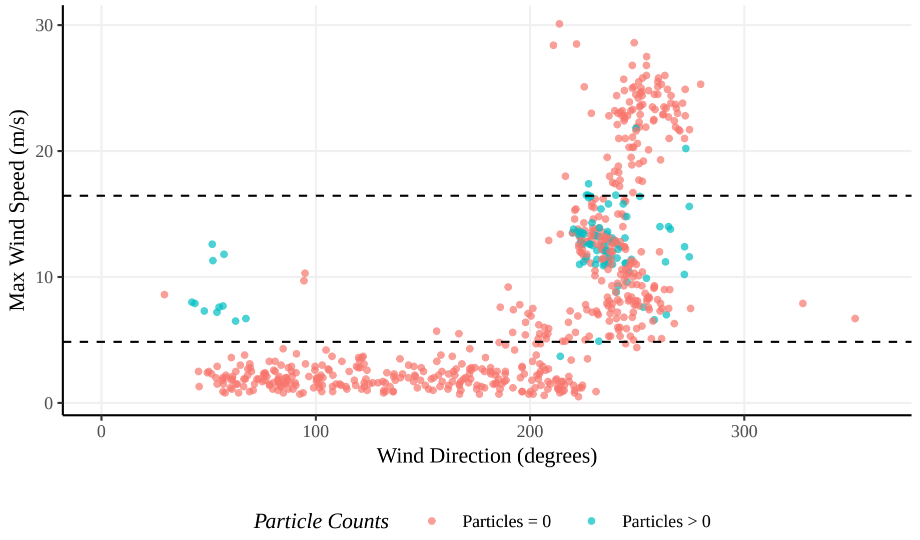
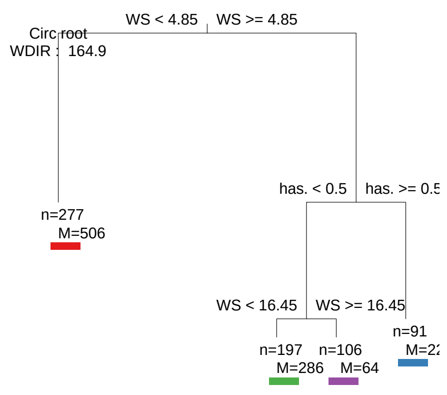
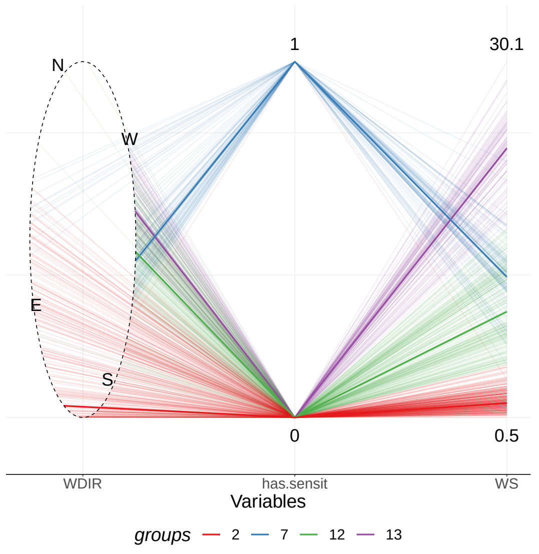


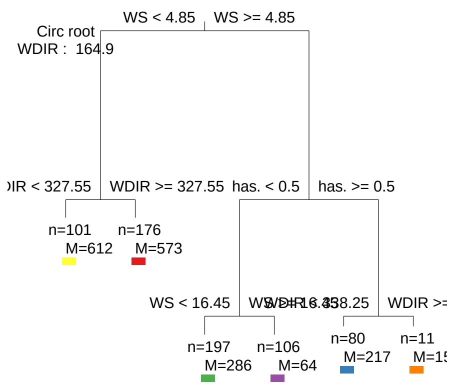
Figure 3.9: Scatterplot of wind direction and wind speed with the particle presence/absence indicated by color in August 4–8, 2007 data. Dashed lines are the splits on *WS* depicted in the splitting rule tree in Figure 3.10.

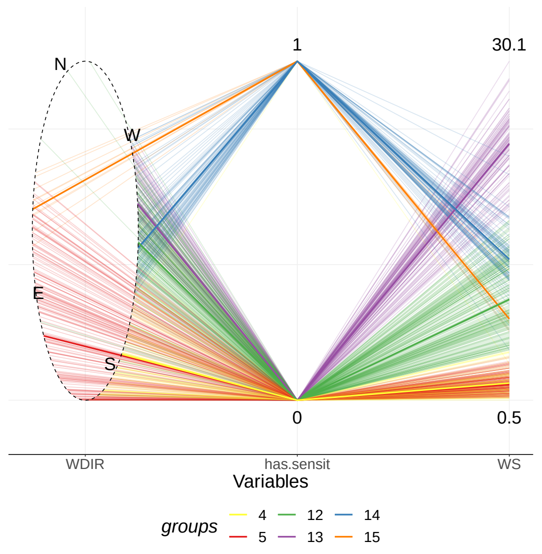
can easily see that winds with particles all belong to a cluster although the split related to the *has.sensit* variable (whether there is particle in the wind) only happens at a later partition, indicating that particles were only detected when the wind was strong enough (> 4.85 m/s). The particles present in low wind speeds may have been too small in size and/or velocity to create enough energy to trigger the sensor since it is force-based. The *has.sensit* and *WS* variables in the PCP clearly show that the characteristics of the first four created clusters can be based solely on these two variables. Partitioning the winds into more or less East and West directions creates two more clusters in the group that has particles and a group of low wind speed observations.





(a) Decision tree for the four-cluster solution. (b) PCP, colors are matched with the tree on the left.





(c) Decision tree for the six-cluster solution. (d) PCP, colors are matched with the tree on the left.

Figure 3.10: Monothetic clustering results of particle count data from August 4–8, 2007.

Table 3.1: Explanation of the four cluster solution in August 4–8, 2007. Results for the cluster Medoid (M), means (m), and proportion of observations with particles (p(Particles)) are presented.

Cluster ID	n	Splitting Rule	Particle M	p(Particles)	WS M	WS m	WDIR M
2	277	WS $<$ 4.85	0	0	1.7	2.0	131
12	197	has.sensit = 0 & WS $\in$ [4.85, 16.45)	0	0	9.3	9.8	238
13	106	has.sensit = 0 & WS $\geq$ 16.45	0	0	22.9	22.8	251
7	91	has.sensit = 1 & WS $\geq$ 4.85	1	1	12.2	12.2	235

Table 3.2: Explanation of the six cluster solution in August 4–8, 2007. Results for the cluster Medoid (M), means (m), and propotion of observations with particles (p(Particles)) are presented.

Cluster ID	n	Splitting Rule	Particle M	p(Particles)	WS M	WS m	WDIR M
4	101	WS < 4.85 & WDIR $\in (165, 327.55)$	0	0.01	2.0	2.0	196
5	176	WS < 4.85 & WDIR $\in (327.55, 0, 165)$	0	0.00	1.9	2.0	100
12	197	has.sensit $= 0$ & WS $\in [4.85, 16.45)$	0	0.00	9.3	9.8	238
13	106	has.sensit $= 0$ & WS ≥ 16.45	0	0.00	22.9	22.8	251
14	80	has.sensit $= 1$ & WDIR $\in (165, 338.25)$	1	1.00	12.8	12.7	236
15	11	has.sensit $= 1$ & WDIR $\in (338.25, 0, 165)$	1	1.00	7.6	8.6	55

In the 2008 data subset, according to the decision tree and splitting rules (Table 3.3), the algorithm picked wind direction for the first split (at 25.8 and 229.9 degrees) to create the two clusters, one for winds coming mostly from the west, and another for winds coming mostly from the east. In part of the wind recordings coming mostly from the west, having particles or not is the criterion to further partition the cluster into Cluster ID 6 (no particles) and Cluster ID 7 (has particles). The winds coming generally from the east are divided by the speed of them. Fast winds ($WS \geq 6.08$) go into Cluster 5 and slow winds ($WS < 6.08$) go into Cluster ID 4. If we go further into a six cluster solution, the additional two splits are both made in the winds coming from the east side of the valley. In the slower wind speed cluster (Cluster ID 4), wind direction is used to cut the wind observations between the east and south directions. In the faster wind speed cluster (Cluster ID 5), the existence of particles is the splitting criterion.

Visual inspections of Figures 3.12b and 3.12d assist us to have a deeper understanding of the clusters for the 2008 data. First of all, the first split on the wind direction implies that the distance between observations is largely influenced by wind directions while WS and (maybe) *has.sensit* variables were now more similar on both directions and were not separated enough to be the first choice, as happened in the 2007 data. Also, it's interesting to see that most of the winds that have particles belong to the winds coming from the west (purple group) and were split to create two smaller clusters inside the west half of the winds. For the winds from the east, because the number of winds with recorded particles is small, $WDIR$ did not become splitting criterion until after it was further split by the wind speed for the low wind group. Visually, very slow winds mostly came from the south (red group). Winds from the west were among the fastest in speed and also contain most of the particles.

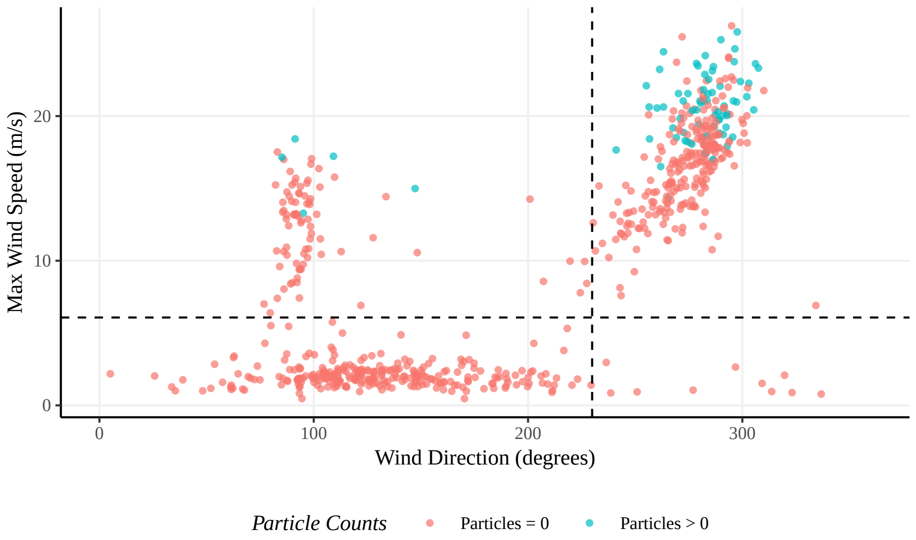
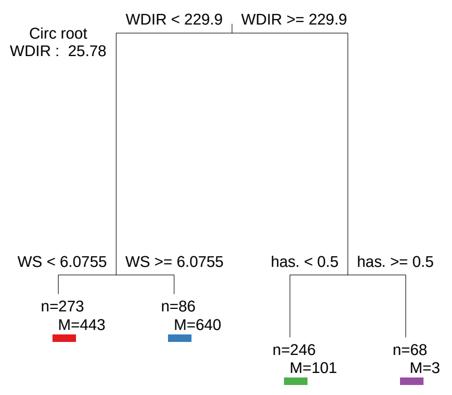
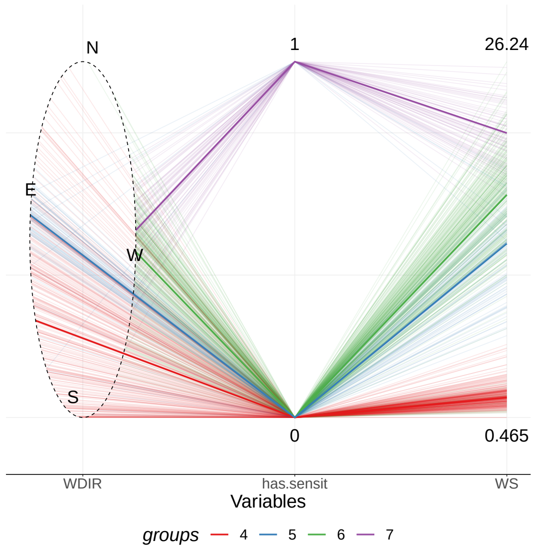


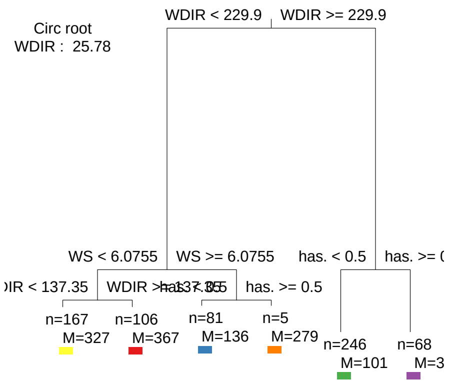
Figure 3.11: Scatterplot of wind direction and wind speed with the particle presence/absence indicated by color in August 4–8, 2007 data. Dashed lines are the splits on *WS* depicted in the splitting rule tree in Figure 3.12



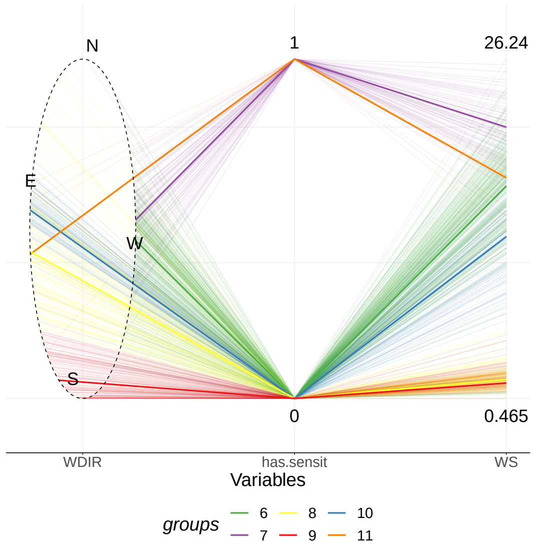
(a) Splitting rule for the four-cluster solution



(b) PCP, colors are matched with the tree.



(c) Splitting rule for the six-cluster solution.



(d) PCP, colors are matched with the tree.

Figure 3.12: Plots for monothetic clustering results of particle counts data from July 7–14, 2008.

Table 3.3: Explanation of the four cluster solution in July 7–14, 2008. Results for the cluster Medoid (M), means (m), and proportion of observations with particles (p(Particles)) are presented.

Cluster ID	n	Splitting Rule	Particle M	p(Particles)	WS M	WS m	WDIR M
4	273	WDIR $\in$ (25.8, 229.9) & WS $<$ 6.0755	0	0.00	1.9	2.1	128
5	86	WDIR $\in$ (25.8, 229.9) & WS $\geq$ 6.0755	0	0.06	13.1	12.5	93
6	246	WDIR $\in$ (229.9, 0, 25.8) & has.sensit = 0	0	0.00	16.6	15.9	277
7	68	WDIR $\in$ (229.9, 0, 25.8) & has.sensit = 1	1	1.00	21.1	20.7	284

Table 3.4: Explanation of the six cluster solution in July 7–14, 2008. Results for the cluster Medoid (M), means (m), and proportion of observations with particles (p(Particles)) are presented.

Cluster ID	n	Splitting Rule	Particle M	p(Particles)	WS M	WS m	WDIR M
8	167	WDIR $\in$ (25.8, 137.35) & WS $<$ 6.0755	0	0	1.9	2.1	108
9	106	WDIR $\in$ (137.35, 229.9) & WS $<$ 6.0755	0	0	1.6	2.0	164
10	81	WDIR $\in$ (25.8, 229.9) & WS $\geq$ 6.0755 & has.sensit = 0	0	0	12.8	12.2	94
11	5	WDIR $\in$ (25.8, 229.9) & WS $\geq$ 6.0755 & has.sensit = 1	1	1	17.2	16.2	109
6	246	WDIR $\in$ (229.9, 0, 25.8) & has.sensit = 0	0	0	16.6	15.9	277
7	68	WDIR $\in$ (229.9, 0, 25.8) & has.sensit = 1	1	1	21.1	20.7	284

3.5 Discussions

Data sets with circular variables have interesting directional information that are useful in many applications. Conceptually, monothetic clustering only needs an extra cut at the circular variable to separate two arcs compared to conventional quantitative variables. This unsupervised method was able to detect interesting underlying structure in the data. Monothetic clustering, specifically, with its ability to create a decision tree showing the shared characteristics of the data points in a cluster, can show the relationships among examined variables. Also, the ability to work on mixed variables including circular variables and categorical variables enhances the flexibility of applications of monothetic clustering while preserving the interpretability of the clusters without compromising the natural characteristics of a circular variable.

Visualizations of cluster results are important to assist in interpretation and exploration of characteristics of the clusters. PCPs showed their flexibility in plotting many variables concurrently and keeping the relative position of values in clusters. Displaying circular variables as ellipses in PCPs retains most of the features of the variables and does not create strange properties as happened in the trigonometric transformations or treating them "linearly". The ability to rotate the values on the ellipse is an improvement to the original version of PCPs and gives researchers flexibility to reduce the overlapping of observations' segments. This idea can be extended to other types of variables such as categorical variables where categories can be sorted to achieve the same effect. As an extension of the PCPs with circular variables, a 3-D version of the plot, where users can interactively rotate the ellipses, change the order of variables, and move the whole plots to have various perspectives would be much more useful and should be considered in the future. However, R (R Core Team, 2019), the language currently used for the implementation of monothetic clustering and visualization, is still somewhat limited in 3-D visualization to realize that idea. The flattened 3-D presentations employed here retain the abstract visualization of a circle with an ellipse shape and helps separate lines coming from different angles, but may require users' involvement to make some rotations to be able to clearly show some observations.

The clustering of the Antarctic particle count data showed that the monothetic divisive method can provide interesting results based on the shared characteristics of wind clusters. The results not only confirm the relationships between pairs of variables found in the original paper, but go beyond by exploring the multivariate relationships between wind speed, wind direction, and the existence of particles. We can assess many characteristics of the winds in the McMurdo Dry Valleys by looking at the decision tree of the resulting clusters. Specifically, the *WDIR* values used in the decision trees reflect the bimodal nature of the winds (down-valley föhn winds and up-valley sea breezes), with the presence of particles and large wind speeds characteristics of some clusters showing that the particles were associated with strong winds. Also, rose plots and PCPs of the cluster solutions visualize the relationships between variables, reinforcing the knowledge that stronger winds are associated with winds coming from the west rather than coming from the east. This means that the föhn winds are stronger than the sea breezes in the examined time frame (July and August which are in the winter in Antarctica).

There is room to extend the monothetic clustering on circular data developed here. Other distance metrics for circular variables can be considered in the `MonoClust` function (see Chapter 5 for details) and it may be of interest to explore and compare these results to each other. Currently, the algorithm searches all pairs of arcs in clock's hour and minute hands to select the pair that has the largest improvement and this is not computationally effective. Moreover, until circular variables are used in a split and the arcs can be regarded as conventional variables further down the splitting tree, the “clocking” method needs to be done in every split, which makes the algorithm slower. It may be possible to improve this by using other search algorithms that can stratify the values and avoid getting stuck at local optima. Additionally, the results of best arc searches can be stored and used later when the circular variables become candidates in later splits. This work is left for future research and explorations.

CHAPTER FOUR

CLUSTERING ON FUNCTIONAL DATA

Contribution of Authors and Co-Authors

Author: Tan V. Tran

Contributions: Responsible for the majority of the writing.

Co-Author: Mark C. Greenwood

Contributions: Provided guidance and feedback on statistical analysis and drafts of the manuscript.

Manuscript Information

Tan V. Tran and Mark C. Greenwood

PeerJ – the Journal of Life and Environmental Sciences

Status of Manuscript:

__X__ Prepared for submission to a peer-reviewed journal

_____ Officially submitted to a peer-reviewed journal

_____ Accepted by a peer-reviewed journal

_____ Published in a peer-reviewed journal

Published by PeerJ

Abstract: The Arctic sea ice extent and area have been recorded since November 1978 until today by the National Snow and Ice Data Center and are available to the public as nearly daily data. There are studies considering the data set as either time series data or uncorrelated measurements of each day of the year. In this research, we complement that view by considering the ice extent data as functional data with functions estimated for each year and exploring cluster analysis of the yearly functions. Two modifications of monothetic clustering, a type of clustering that creates clusters that share common characteristics, are proposed. One uses daily estimates to cluster the curves and another divides the functional data into subregions and uses the information in those subregions to partition the data set. The proposed clustering techniques can also limit their searching variables/subregions so they are able to classify the curves based on the results from just a portion of the curve, say the first few months of the year. To illustrate the methods, one random year in each decade is withheld to use as a test data set for the clustering results created by the other years to assess the classification performance of the methods.

4.1 Introduction

Measurements y taken over some ordered index t such as time, frequency, or space and are thought of as curves or functions of the index t are called *functional data*, $y(t)$ (Ramsay and Silverman, 2005). Functional data have, possibly, a high frequency of observations over the index t and are assumed to be generated by a smooth underlying process. Some examples of functional data include the growth curves for girls in the Berkeley Growth Study (Tuddenham and Snyder, 1954), hydraulic gradients in wetlands (Greenwood et al., 2011), or ice extent area over time in the Arctic Sea (Fetterer et al., 2018). The last example is the motivational data set for this study and takes a new approach with these data.

Clustering can be useful for functional data to find groups of curves sharing common characteristics and to find representative curves corresponding to different modes of variation in the data. Early applications of clustering functional data considered the functions at arbitrary, or originally observed, discrete values of the argument t, which does not fully capitalize on the functional nature of the data. More recently, functional data clustering has had more attention and various approaches to clustering functional data have been proposed (Hitchcock and Greenwood, 2015).

Functional data are usually represented as a combination of basis functions, such as splines, Fourier basis, or Wavelets, and their estimated basis coefficients. Tarpey and Kinateder (2003) used k-means (MacQueen, 1967) on these basis representations which narrow downs to clustering the estimated coefficients when the splines used are orthogonal. In the same paper, they also showed two interesting results. One is if the variability of shapes of curves is of interest, the clustering algorithm can be performed on the first derivative of the functions to effectively remove the variability of the means of the functions. Besides, they proved that clustering the basis coefficients is correct only if the chosen basis functions are orthogonal over the interval T. Other researchers have attempted to adapt multivariate clustering methods to functional data. Distance-based clustering techniques such as Ward's hierarchical method and k-means have been used on dissimilarity matrices from pairs of functions. Tarpey (2007) used L_2 distance between functions, $y_i(t)$ and $y_j(t)$, with t is the index within a fixed interval T, defined as:

$$d(y_i, y_j) = \sqrt{\int_T [y_i(t) y_j(t)]^2 dt} \tag{4.1}$$

as a distance measure between functions and their cluster mean in the k-means clustering framework. He showed that using different basis functions (hence different linear transformations) to represent functions can result in very different cluster results. Therefore, this approach is more effective when some linear transformation is performed on the curves to minimize the within-cluster variance and maximize the between-cluster variance. Suarez and Ghosal (2016) suggested calculating dissimilarity measurement based on the average number of shared detail coefficients in the Wavelet basis representation of the functions. This approach involves using Dirichlet process prior distributions to the detail coefficients and estimating them by the Monte Carlo Markov Chain (MCMC) method. Model-based clustering (Banfield and Raftery, 1993) is also used in functional data clustering. James and Sugar (2003) further parameterized the function's basis coefficients to form a random effects model to cluster sparsely sampled functional data in a model-based manner. Other functional data clustering approaches involve projecting sparsely measured functional data into a reduced-dimensional subspace by applying functional principal component analysis (FPCA) and applying clustering techniques on the FPCA scores (Zajacova et al., 2015) or using rank-based correlation as the dissimilarity measure for functional data (Heckman and Zamar, 2000). An extensive overview of various functional data clustering research can be found in Hitchcock and Greenwood (2015).

The National Snow & Ice Data Center has been collecting daily Arctic sea ice extent since October 1978 to present (Fetterer et al., 2018). The data are publicly available (`https://nsidc.org/data/G02135/versions/3`) and have been analyzed in many places. Most have observed a decrease in the ice extent in Arctic and then have tried try to find evidence of a relationship of this phenomenon to various factors, such as polar winds (Serreze et al., 2003) and greenhouse gas (Serreze et al., 2007). Other researchers used just a portion of these data, for example, using the sea ice extent in the month of March from 1982–2007 to explore the relationship between the mean ice extent cover and mean ice thickness in spring over the years (Maslanik et al., 2007). They concluded that the loss of older Arctic ice continued with its lowest point in the summer of 2007 (the most recent year in their study). Our research has not shown any studies that have considered the ice extent as functional data or daily values, or attempted to apply cluster analysis methods to detect underlying patterns of the yearly ice cover and how those may have changed over time.

In this study, we apply monothetic clustering and a similar clustering method that utilizes the combined information of variables in intervals of the functional data called *Partitioning Using Local Subregions* (PULS) on the Arctic sea ice extent data to show that cluster analysis can give interesting and similar conclusions as other studies and goes beyond what other studies provided by giving us the ability to characterize the similarities and differences of the yearly ice patterns from 1978 to 2018 based on days or months information.

4.2 Arctic Sea Ice Extent as Functional Data

The Arctic, the region around the north pole, is geographically different from the Antarctic mentioned in Chapter 3 in that it is an ocean surrounded by the continents, instead of a continent itself (Figure 4.1). Therefore, the sea ice area in the Arctic is the subject of much research involving how changes in sea ice may be affected by climate factors such as greenhouse gases, surface oscillations, and seasons (Maslanik et al., 2007; Serreze et al., 2003; Serreze et al., 2007).

The National Snow & Ice Data Center (NSIDC) recorded the daily ice extent by adding up the areas of satellite data cells that are ice-covered, defined by nominally 25 km $\times$ 25 km grid cells having greater than 15 percent ice concentration. There have been several instruments used to get the satellite passive microwave brightness temperatures for the ice extent data over the course of the study. From October 26th, 1978 to August 20th, 1987, the data were collected every other day using the

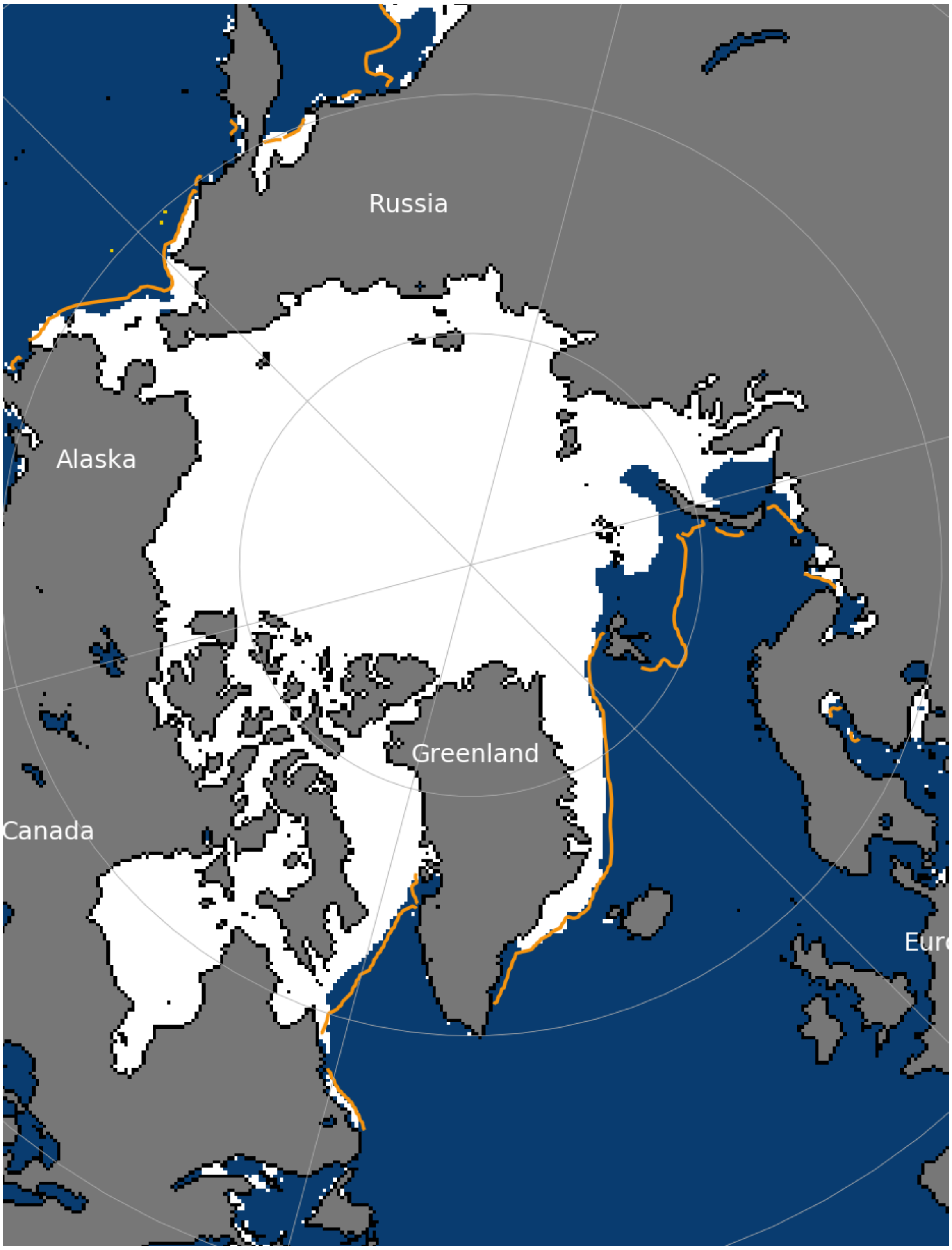


Figure 4.1: Arctic ice from satellite on December 25, 2018. The yellow line is the median ice edge from 1981–2010. Imagery from the NASA MODIS instrument, courtesy NASA NSIDC DAAC.

Nimbus-7 SMMR instrument. After that, a series of SSM/I and then SSMIS were used and upgraded regularly and the measurements were changed to being available daily. There are no data from December 3rd, 1987 to January 13th, 1988 due to satellite problems (Fetterer et al., 2018). The observation times in the data over the years are shown in Figure 4.2. The ice extent data fit the characteristics of functional data in which the ice extent is a function of days of years as they are intensively sampled and appear to be relatively smooth over the argument (here, day of year, Figure 4.3).

To apply cluster analysis to this data set, we removed the years that had missing measurements in many consecutive days, namely, 1978 (data were not available until October) and in 1987 and 1988 (because of the satellite problems). We also randomly withheld one year in each decade for validation purposes. They are 1982, 1990, 2006, and 2018. However, a part of the data from those years was used in the smoothing process of other years (to be explained in the next paragraphs).

To use discretely measured observations, $y_{i1}, y_{i2}, \ldots, y_{in}$, for day 1 to estimate day n in year i to a functional datum, $f(t)$, computable for any value t in 1 to 366, a *smoothing* process needs to be performed. In order to do that, a basis function system is first defined as a linear combination of basis functions, $\phi_k(t)$, which are used to estimate a smoothed version of the functional datum,

$$\widehat{f}(t) = \sum_{k=1}^{K} \widehat{c_k}\phi_k(t), \tag{4.2}$$

where $\phi_k(t)$ are known functions that are mathematically independent of each other and K needs to be large enough so that the set of the basis functions with their coefficients, $\widehat{c_k}$, can adequately approximate the true function, $f(t)$. Commonly used basis functions for $\phi_k(t)$ include Fourier series, Wavelets, or a spline basis system.

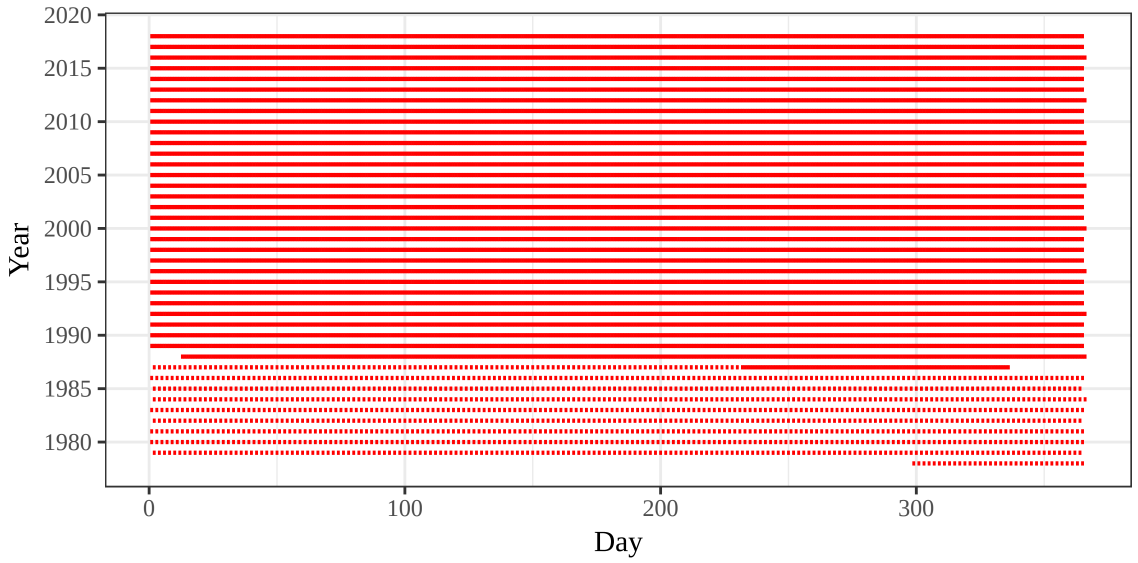


Figure 4.2: Time of measurements of Arctic sea ice extent over the years. From October 1978 to August 1987, the measurements were taken every other day. After that, they were measured daily. There was a gap at the end of 1987 due to satellite errors. The most recent data point in this plot is on December 31, 2018.

A spline basis system splits the index into intervals and in each interval, a linear combination of polynomials is used to represent the data in that interval. Moreover, the functions are defined so that the derivatives of the functions are continuous at the boundaries of the intervals (called *knots*). One of the most popular spline basis systems is the set of so-called *B-splines* developed by De Boor (1972). A B-spline basis system is determined by the order m of the polynomials used and the knot sequence τ which determines transition points for bases (both number and location). One of the biggest advantages of the B-spline system over others is that it is computationally efficient and readily available in statistical software, such as R (R Core Team, 2019). More details of B-splines can be found in Ramsay and Silverman (Chapter 3, 2005).

For the Arctic ice extent data, despite working with years as the functional observations, continuity between the last day of one year and the first day of next year should be maintained. Therefore, we attach previous year's December at the

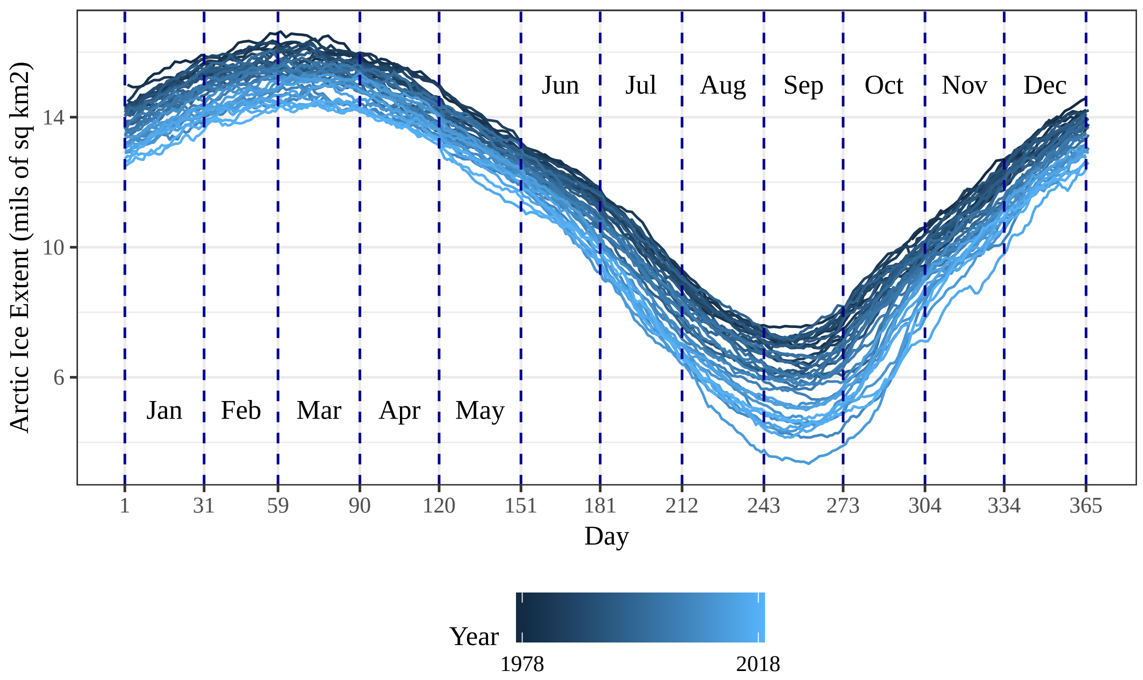


Figure 4.3: The Arctic sea ice extent from 1979 to 2018. Each year is a curve connecting available ice extents.

beginning and next year's January at the end of each year, padding it with extra observations, then estimate functional data using each year with extended months. Only $f(t)$ from January 1 (day = 1) to December 31 (day = 366) are used in analyses. To ensure the continuity of at least two derivatives of the functions, a B-spline basis system $B_k(t, \tau)$ with the order of $m = 4$ (or the degree of 3, which is a cubic spline) was created (an excerpt of the B-spline bases on the first 10 days of a year is depicted in Figure 4.4). The smoothed function for a single year, $\widehat{f}(t)$, in Eq. 4.2 is formed by a linear combination of the B-spline basis functions and their coefficients as

$$\widehat{f}(t) = \sum_{k=1}^{m+L-1} \widehat{c}_k B_k(t, \tau),$$

with $L - 1$ equal to the number of knots and t defining the locations of those knots.

To estimate the coefficients c_k for each basis function, on the one hand, we want to make sure that the estimated curve is a good fit to the data by reducing the residual sum of squares, $\sum_{t=1}^{T}[y_{it} - \widehat{f}_i(t)]^2$ for the i-th functional observation. On the other hand, fitting the original points too well makes the curves overly "wiggly", contradicts the purpose of smoothing, and includes the noise in the estimated function. Therefore, penalized sum of squared errors, $PENSSE_\lambda(\widehat{f_i(t)}|y)$, is used to estimate the function with a control for the roughness of the estimated functions, $\widehat{f(t)}$. It is written as

$$PENSSE_\lambda(\widehat{f_i(t)}|y) = \sum(y_{it} - \widehat{f(t)})^2 + \lambda \times PEN_2(\widehat{f}),$$

where $PEN_2(\widehat{f})$ is a measure of function's roughness, which here relates to the second derivative, $D^2\widehat{f}_i(t)$, of the estimated function

$$PEN_2(\widehat{f}_i) = \int [D^2\widehat{f}_i(s)]^2 ds,$$

and the smoothing parameter, λ, determines the weight placed on smoothness of the curve. When λ is zero, there is no penalty for the roughness, thus using ordinary least squares. As λ increases, $PENSSE_\lambda(\widehat{f}_i)$ considers the smoothness of $\widehat{f}_i$ more. The smoothing parameter λ can be selected using cross-validation by withholding a part of the data as a *validation sample* and estimating the model with the rest of the data, called the *training sample.* This procedure is repeated and the resulting error of sum of squares is calculated for each value of λ. The value that yields its minimum is chosen. It is more common to use generalized cross-validation (GCV, Craven and Wahba, 1978) to select smoothing parameters in FDA because it can be calculated directly from the residuals of the original model and it has been found to be more reliable than regular cross-validation (Ramsay and Silverman, 2005). GCV is also often utilized in Generalized Additive Models (Wood, 2006). The GCV criterion has

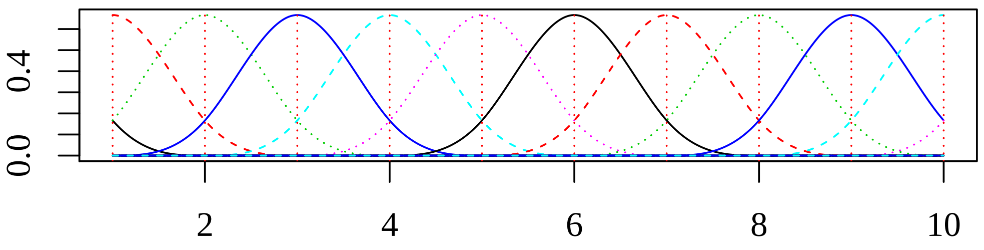


Figure 4.4: The basis functions of the order four spline and knots at every day (vertical lines) used to smooth the Arctic ice extent data, showing the first 10 days of the year (after attaching Dec of last year and Jan of next year, making 425 days for this example year). Because of that "padding", there is no boundary behavior at the edges of functions.

the formula

$$GCV_g = \frac{\sum_{t=1}^{T}(y_{it} - \hat{y}_{it})^2}{[1 - \sum A_{tt}/n]^2}, \tag{4.3}$$

where A_{tt} are the diagonal values of the hat matrix A that can be used to obtain $\widehat{y} = Ay$ and incorporates both the basis functions and the smoothing penalty.

In the Arctic sea ice extent data, GCV optimization is used to estimate λ for each extended year. years 1982, 1990, 2006, and 2018 for validation purposes. The smoothed ice extent data can be seen in Figure 4.5 and a comparison between two raw yearly observations and smoothed curves can be seen in Figures 4.6 and 4.7. They show minimal impacts of the smoothing process in estimating the functional from the original discretely observed ice extent values.

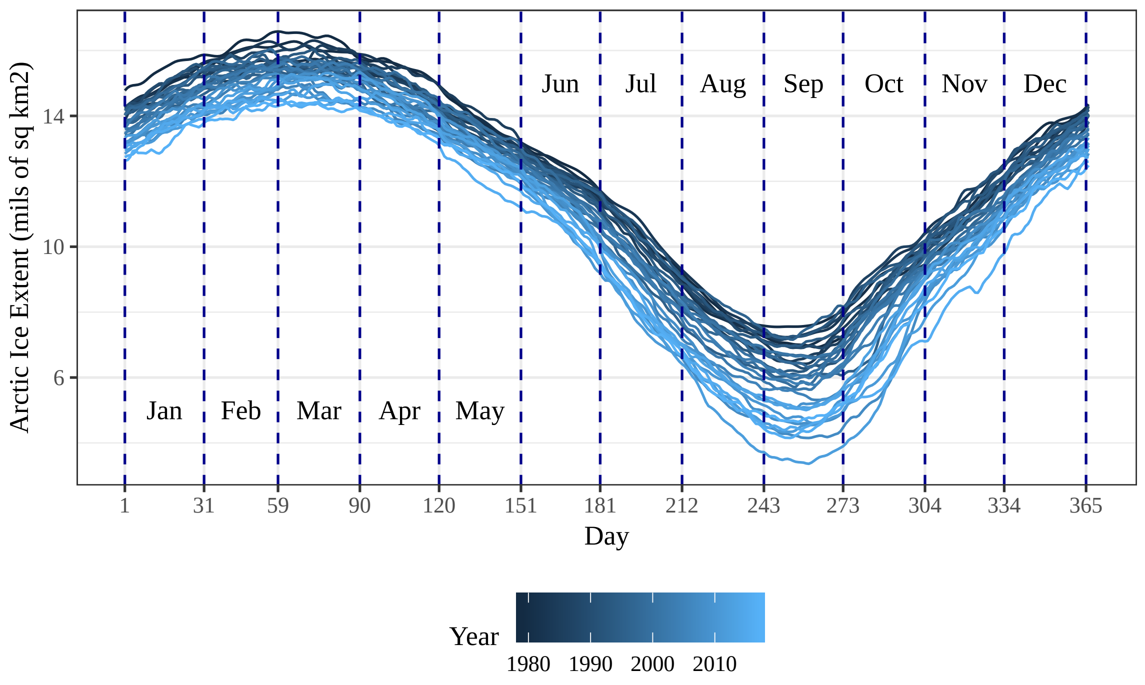


Figure 4.5: Smoothed functional data for Arctic Sea Ice extent from 1979 to 2017 (excluding 1978, 1987-88 because of missing data, and 1982, 1990, 2006, and 2018 for validation purposes).

4.3 Clustering Ice Extent Data: Monothetic Clustering & Partitioning Using Local Subregions (PULS)

Monothetic cluster analysis, as proposed in Chavent (1998), is a clustering algorithm designed to optimally partition a data set using recursive binary partitions to optimize the inertia change at each split. In multivariate (non-functional) data with discretely observed y_{iq} with $q = 1, \ldots, Q$ variables, the inertia for a cluster is defined as the total variability around the cluster centroid. In the case of the L_2-norm, the squared Euclidean distance is used as the inertia for cluster k, $I(C_k)$, defined as

$$I(C_k) = \sum_{i \in C_k} \sum_{q=1}^{Q} (y_{iq} - \overline{y}_{\cdot q})^2, \tag{4.4}$$

where $\overline{y}_{.q}$ is the mean of all observations y_{iq} in variable q. The objective of the monothetic algorithm when splitting cluster C_k is to maximize the difference, or change, in inertia between C_k and the new sub-partitions C_{kL} and C_{kR},

$$\arg\max(I(C_k) - I(C_{kL}) - I(C_{kR})), \tag{4.5}$$

and the same method is then recursively applied to each sub-partition.

Because the inertia calculation involves Euclidean distance between discretely observed (in time) observations, it still uses the Arctic ice extent data in its discrete form (numerical extent on each day, y_{iq}) instead of explicitly using its functional form. However, as shown above, the data were recorded at different frequencies over the years (every other day before 1987 and daily after that) and the number of observations is not the same in every year (some years have 365 days while others have 366 days). Although monothetic clustering does not work directly on the functional form of the Arctic ice extent data, transforming the data into its functional representations and then exploiting the properties of those continuous functions to discretize the values to every day of a year can help overcome the problem with different sampling rates in the data set. Moreover, Hitchcock et al. (2007) showed via simulations that pre-smoothed functional data are often clustered more correctly using PAM and Ward's method than the original discretely observed points, so it is possible that the same process could help monothetic clustering as well. Specifically, after estimating smoothing using penalized B-splines as described in the previous section, the resulting functions can be used to interpolate the missing measurements before 1987. All functional data are evaluated for days $i = 1, \ldots, n$, where $n = 366$ days in each year to make the data balanced across the years. After 1987, all of the measurements can be predicted from the functional data to potentially reduce

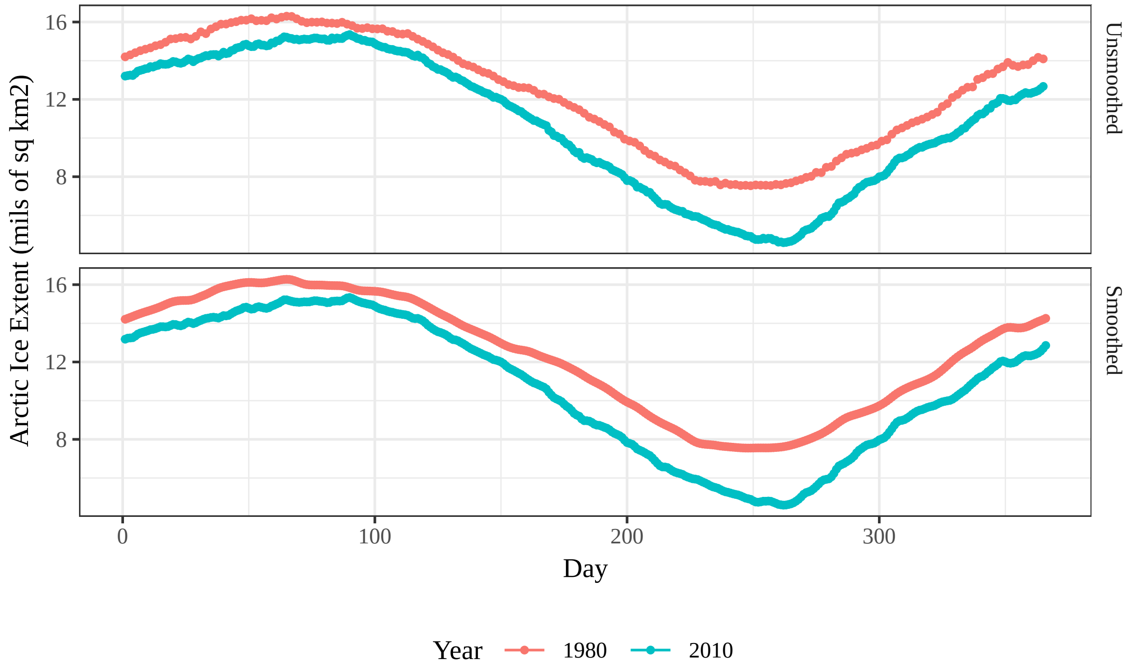


Figure 4.6: Comparison between discrete observations over time and smoothed curves for years 1980 and 2010. The smoothed data are less wiggly. Year 1980 was originally measured less frequently than the later years (every other day), thus the smoothing is slightly more pronounced.

measurement errors slightly and still retain the original dimensions of the data set (366 observations per year). Figure 4.7 shows the smoothing results for January 1980 and 2010. We can see that the solid lines (functional data) are smoother than the dotted lines (raw data) but retain the overall original shape and the missing observations every other day for years before 1987 are reasonably interpolated, with similar patterns observed throughout the data set.

The application of monothetic clustering to functional data is not without drawbacks. First of all, it does not utilize the functional properties besides smoothing and imputation to obtain the discretely observed data. Also, 366 days in a year are 366 variables that need to be explored for the best split at each partitioning step,

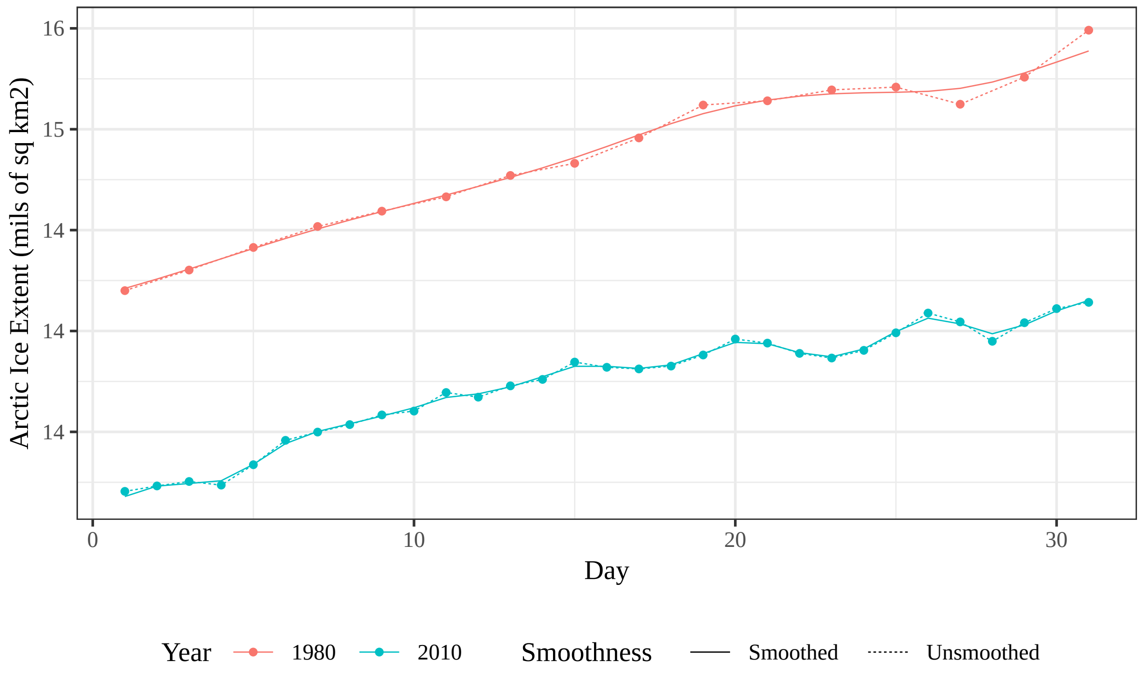


Figure 4.7: Comparison between discrete observations over time and smoothed curves for January in 1980 and 2010. By considering this short time window of 31 days, differences between raw observations and smoothed functional can be observed, especially for 1980 when measurements were taken every other day.

which slows the algorithm down, exposing a computational challenge in monothetic clustering with very wide data. Last, but not least, in curves, it is expected that the neighboring time points (days, or more generally, variables) are not very different from each other and may provide the same or similar split information. When the algorithm encounters more than one candidate splitting variable resulting in the same sub-partition structure, the algorithm must arbitrarily pick one variable (currently, the algorithm is set to pick the earliest date/first column in that group of candidates), ignoring all other candidates. This is unfortunate for interpretation of clusters since a variety of days can all imply the same groups but we would only know of one day determining the cluster splitting rule and not know that similar differences existed across many different days.

4.3.1 Partitioning Using Local Subregions

In some functional data applications, there is pre-existing knowledge of regions of interest. It could be intervals of time that are known to be very different or define important features of typical curves, or, more generally, sets of variables that most clearly define common groups. For example, for functional data estimated from spectral responses, the different frequency bands of the electromagnetic spectrum (visible vs. red edge vs. infrared) have been shown to be discriminative by Vsevolozhskaya et al. (2014). This method is somewhat similar to sparse clustering in terms of using several variables to group the observations (Witten and Tibshirani, 2010) when the number of variables is much larger than the number of observations. However, the basic difference is that this method uses pre-existing knowledge to choose the subregions as splitting candidates while sparse clustering uses unsupervised methods to update the weight parameters to remove variables from the dissimilarity matrix deemed to be unimportant or not showing clear cluster structure. Often, researchers can identify the regions of interest in functional data prior to the analysis. However, in time series data like the Arctic ice extent data, where there are no specific intervals of time known *a priori* to define differences in each year's behavior, we can define subregions as a large number of equal sized intervals that are mutually exclusive and exhaustive. This approach was also discussed in Vsevolozhskaya et al. (2014). Here, months of years where the lengths are similar and have practical meaning for later interpretation are used to facilitate interpretation of the results.

After defining the mutually exclusive R subregions $[a_1, b_1], [a_2, b_2], \ldots, [a_R, b_R]$ of interest, the L_2 distance between all pairs of functions $y_i(t)$ and $y_j(t)$,

$$d_R(y_i, y_j) = \sqrt{\int_{a_r}^{b_r} [y_i(t) y_j(t)]^2 dt},$$

is calculated to obtain a dissimilarity matrix for each of the R subregions. Adapting the idea of monothetic clustering, each subregion is separately considered as a splitting candidate for the next 2-group partitioning, using any commonly used clustering technique such as k-means (MacQueen, 1967), Ward's agglomerative hierarchical method (Ward, 1963), monothetic clustering, or partitioning around medoids (PAM, Kaufman and Rousseeuw, 1990). PAM is a clustering technique that picks an existing observation (curve) in a cluster as its representative, called *medoid*, instead of using an artificially constructed *centroid* as in the k-means technique so can be less impacted by outlying observations.

Inertia (Eq. 4.4) is again used as the global criterion. Among the R candidates (one from each subregion), the split having the largest decrease in inertia is chosen as the best split (Eq. 4.5). The algorithm is then recursively applied to the resulting sub-partitions until a specified number of partitions is reached or a stopping rule is met (see Chapter 2 for more details).

This algorithm (which we name *PULS* for Partitioning Using Local Subregions) overcomes some disadvantages of monothetic clustering. It better utilizes functional data properties to find the dissimilarity matrix and uses the combined information from pre-existing knowledge of "important" subregions or levels of aggregation in the functional data when considering the next partitioning. It explores fewer possible partitions and aggregates more information in each split considered, possibly decreasing the rate of tied splits and possible "false" or spurious splits.

4.3.2 Constrained Version of Monothetic Clustering and PULS

In a data set, not all variables are created equal. Some variables are easier or cheaper to measure than others. For example, a data set on human vital information may include body fat percentage, height, and weight. Height and weight are much easier to obtain accurate measurements for than body fat percentage. If the data can be partitioned using only these easy-to-measure variables and are comparable to the results made by using all variables, it can help reduce effort and cost in the future studies. Other potential applications include clustering snow and site characteristics including snow density, where rules based on GIS derivable variables including aspect are used to partition the measured areas (such as for data like that collected in Wetlaufer et al., 2016), or the prediction of eBay auctions' final prices can be constrained to use information at the beginning or at the end of the auction to form clusters (Wang et al., 2008). Another application is for this ice extent data set, where information from some months or seasonal times of the year can be used to reasonably partition the curves. A potential benefit of that constraint is when ice extent just from the beginning of a year could be used to predict the cluster that year will fall into. Generally, using only portions of functional data to cluster curves may be of most interest with time series versions of functional data to provide early predictions of group membership for new functions that were not used to develop the clusters.

Both monothetic clustering and PULS algorithms implemented in the **monoClust** and **PULS** R packages have options to limit the splitting candidates to a subset of variables (in case of monothetic) or subregions (in case of PULS). Besides the benefits of reducing effort and cost in future studies, the constrained versions of the algorithms also speed up the running time, making the cluster estimation faster and more efficient.

4.4 Results

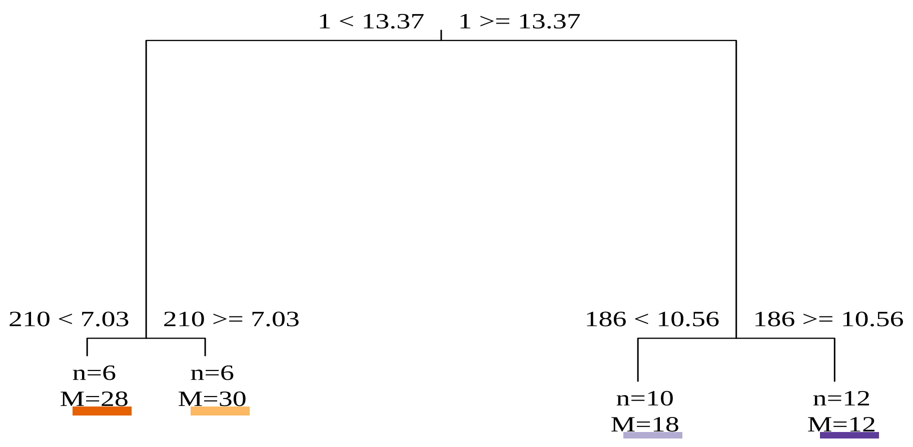
In this study, we are seeking for years that have similar patterns and use the splitting rules of monothetic clustering and PULS techniques to uncover the characteristics that are shared (both timing and patterns) within clusters and differences between clusters. Full calendar years are used as observations; years that have missing data for a continuous length of time, including 1976 (when data were first recorded in October), 1985, 1986 (there are missing data points at the end of 1985 to the beginning of 1986 because of satellite problems), and 2019 (data are not complete at the most recent data update), are removed from the data set. The B-spline basis system with cubic polynomials (order 4) is used to represent the Arctic ice extent data as functional data and data are padded at the beginning and end of years to reduce edge effects in estimating the functional data.

After obtaining a functional representation of the ice extent as a function of day of year, in order to apply monothetic clustering, individual missing measurement days before 1987 and day 366 are interpolated (so every year was considered as a leap year with 29 days in February) to create a new data set with smooth, discretely observed sea ice extent measurements for these years. Furthermore, to demonstrate the potential for cluster prediction using monothetic clustering, data from one year in each decade is randomly withheld from applying monothetic clustering, together with the most recent full year 2018, to be withheld from the functional data estimation. Those withheld years are 1982, 1990, 2006, and 2018. This leaves $n = 37$ years for clustering and 4 for "validation".

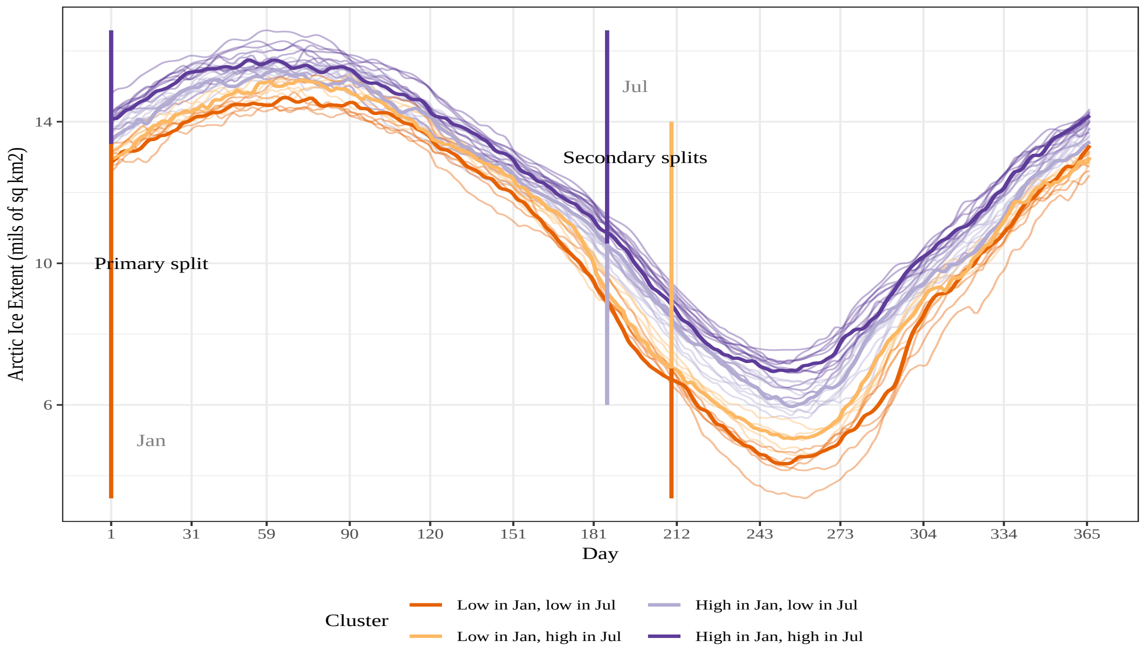
The splitting tree and visualization of the curves are depicted in Figure 4.8. The first split is picked at Day 1 of a year. If the ice extent on Day 1 of that year is greater than 13.37 millions of squared kilometers, it belongs to the right branch, otherwise it is in the left branch. Years with lower ice extent in Day 1 will be further split based on their Day 190 (6th of July) extent values higher or lower than 8.67 millions of squared kilometers. Years at the right branch are split based on Day 186 (2nd of July) at 10.56 millions of squared kilometers. The monothetic algorithm gave warnings at every splitting node that there are more than one splitting candidate at all three nodes on the tree, so Day 1, 186, and 190 are chosen because they all are the *earliest day* in the list of equally "best" candidates and those candidates are usually several days in row. So it is more appropriate to summarize the rules as in the legends of Figure 4.8b using the months instead of the exact days in the splitting tree.

The PULS algorithm works directly on the functional form of the ice extent data. Ward's hierarchical method and PAM were used as the underlying clustering techniques at the subregion. In these data, they both produced the same results. The subregions where the partitioning occurred at every step are plotted in Figure 4.9a together with the highlighted months (Figure 4.9b). The ice extent data in July and August are the source of partitioning. The first split occurs in July and so is the second clustering for years with low extent. For the years with high July extent (right branch of the tree), the next split happens in August, creating two clusters with 13 years and 11 years. Because the underlying clustering technique at the subregion (month) is not monothetic clustering, the splitting tree is not as helpful as the monothetic clustering result in Figure 4.8a. A full functional curves plot with colored cluster members in Figure 4.9b is necessary to understand the characteristics of the resulting clusters.

Table 4.1 lists years in each cluster of two clustering techniques with the medoid years, which can be considered as the representative year of that cluster, in bold. There are several observations can be made in this result. The four clusters are very similar between the two techniques, especially noting that the last two clusters

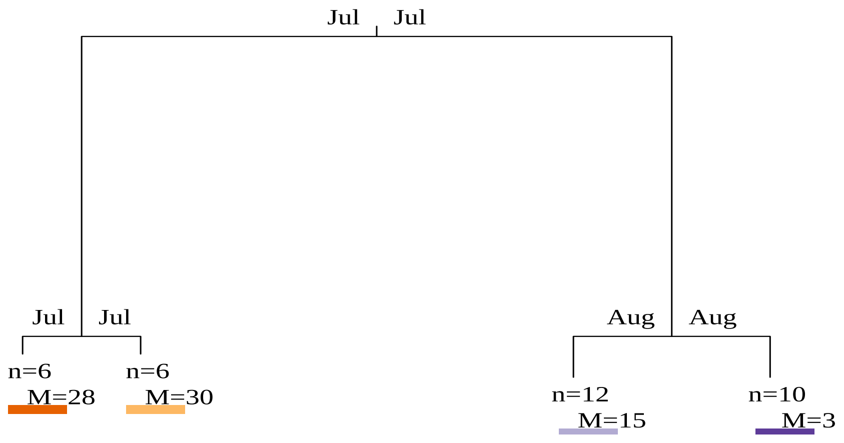


(a) Splitting rule tree at days of years, n is cluster size, M is the cluster medoid (observation's row index).

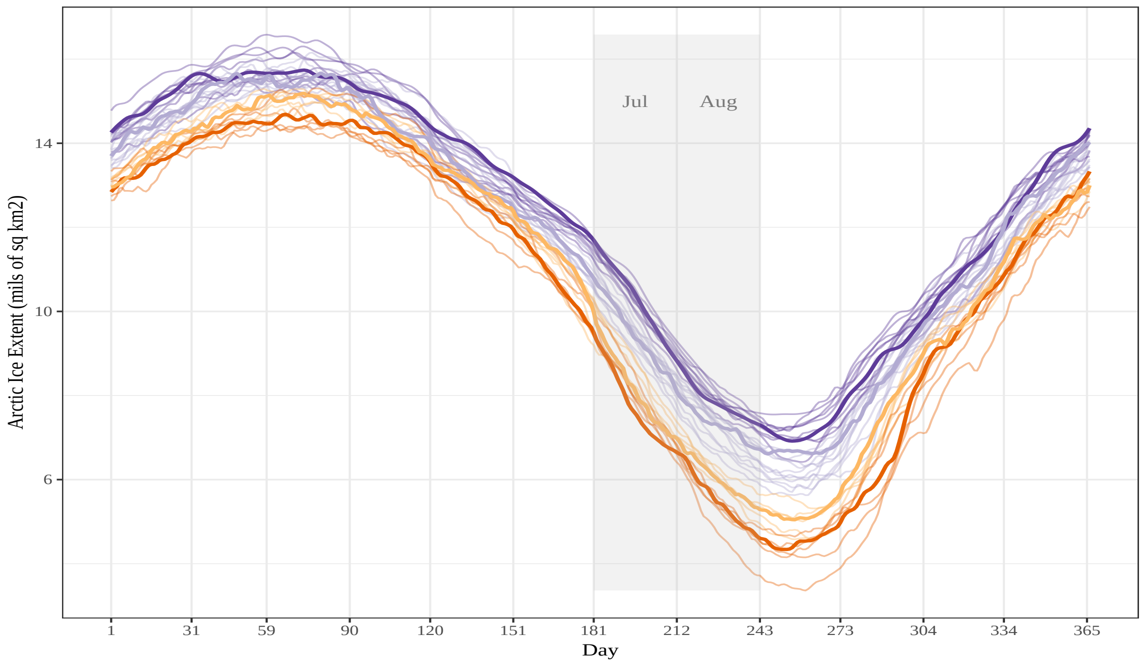


(b) Ice extent as curves with clusters with cluster medoids highlighted and the positions of splits.

Figure 4.8: Four cluster solution using monothetic clustering on Arctic ice extent data for years 1979–1986, 1989–2017 (excluding years 1982, 1990, 2006, and 2018 for prediction purposes).



(a) Splitting rule tree at splitted subregions (months), n is the cluster size, M is the cluster medoid (observation's row index).



(b) Ice extent as curves with clusters with cluster medoids highlighted. The splitting months are shaded.

Figure 4.9: Four cluster solution using PULS on Arctic ice extent data for years 1979–1986, 1989–2017 (excluding years 1982, 1990, 2006, and 2018 for prediction purposes).

are identical. Years 1985 and 1993 belong to cluster 1 in monothetic clustering but to cluster 2 in PULS. Those are the only differences between the two techniques. Consequently, the medoid years are different in clusters 1 and 2 between PULS and monothetic clustering. Because PULS does not create clear splitting rules, we base the interpretation of clusters on monothetic splitting rules. While PULS uses July and August as the splitting subregions, monothetic clustering uses the first day of January for the very first split. However, as mentioned above, the 1st of January is only one among many equally best splitting candidates, and although not shown here, that list includes many days in July and August. Therefore, the two techniques agreed on the partitioning rules and the resulting clusters.

The peak months in winter and summer define the cluster a year belongs to, especially summer time. We can also observe in Figures 4.8b and 4.9b that summer months are when the ice extents are the most different among years. From the top to the bottom of Table 4.1, the ice extent gets lower reflected by the cluster's medoids. There is also a trend to have neighboring years grouped into similar clusters. The pattern in the corresponding clusters indicates that the more recent a year is, the less ice extent it has. This observation agrees with many recent studies where the overall trend of ice extent is decreasing on average over recent time (Maslanik et al., 2007).

We have seen that PULS used July and August as the splitting subregions, so we ran monothetic clustering and PULS again on the same data but this time limited the splitting candidates to only July, August, and September (the summer months in the Arctic). Although not shown here, the cluster results are identical. This solidifies our belief that using those summer months only should be adequate to detect the differences among years and predict the pattern of a new year based solely on these months.

Lastly, using the splitting rules of monothetic clustering, previously withheld

Table 4.1: Years in clusters. The years in bold are the medoid of their cluster. Note that years 1987, 1988, and 2018 were not accounted for because of missing data. Years 1982, 1990, 2006, and 2018 were withheld as testing data.

Name	PULS	MonoClust
High Jan, High Jul	1979–**1981**, 1983–1984, 1986, 1989, 1992, 1994, 1996	1979–1981, 1983–1986, 1989, 1992–**1994**, 1996
High Jan, Low Jul	1985, 1991, 1993, 1995, **1997**–2004	1991, 1995, 1997–**2000**–2004
Low Jan, High Jul	2005, 2008–2010, **2013**, 2014	2005, 2008–2010, **2013**, 2014
Low Jan, Low Jul	2007, **2011**, 2012, 2015–2017	2007, **2011**, 2012, 2015–2017

years 1982, 1990, 2006, and 2018 are assigned into the clusters by looking at the days 1, 186, and 210 of those years (in case of 1982, day 1 data were missing so day 2 was used instead). The cluster prediction is given in Table 4.2. Year 1982 belongs to the cluster with highest ice extents (high in January, high in July), year 1990 belongs to the second highest ice extent cluster (high in January, low in July), 2006 belongs to the second lowest cluster (low in January. high in July) and 2018 belongs to the lowest ice extent cluster (low in January, low in July). The 4 withheld years in 4 decades belonging to 4 different clusters are consistent with the decreasing trend over time of the ice extent in the Arctic area.

4.4.1 Simulation Study

We also set up a simulation study to compare the performance of monothetic clustering and PULS with commonly used clustering techniques where the observations are functional curves and the groups are only distinct from each other in a

Table 4.2: Prediction of withheld years based on the monothetic splitting tree. Day 1 in 1982 was missing so Day 2 was used instead.

Year	Observed Extent	Cluster
1982	Day2 = 14.479; Day186 = 11.344; Day210 = 9.22	High Jan, High Jul
1990	Day1 = 14.319; Day186 = 10.328; Day210 = 7.891	High Jan, Low Jul
2006	Day1 = 13.16; Day186 = 9.218; Day210 = 7.516	Low Jan, High Jul
2018	Day1 = 12.491; Day186 = 9.346; Day210 = 6.815	Low Jan, Low Jan

small subregion of the data. Five clustering techniques include PAM, Ward's method, monothetic clustering, and two versions of PULS, one with PAM and the other with Ward's method as the underlying clustering algorithms.

We generated observations from four true functional curves with the index range from 0 to 100. They were designed to be identical except for the 50–70 interval of the index "time". The data sets were then estimated from the functions to a fine grid at each 0.5 unit of "time". Ten different sets of random noise following a multivariate independent normal distribution are added to the data set to create a total of 40 observations (discretized curves) from four true clusters. Then a smoothing process, similar to what has been done to the Arctic ice extent data, is done, using smoothing cubic B-splines to achieve 40 smoothed functional curves with known true cluster membership (Figure 4.10). The estimated functional curves are also discretized at every "time" unit from 1 to 100 for non-functional clustering methods. There are 1000 such data sets simulated. The five clustering techniques are applied to the simulated data sets using the Euclidean distance metric and its functional adaption in Eq. 4.1, producing four clusters using each method. The true cluster memberships are then used to compare with the cluster results of the five methods to the truth (corrected

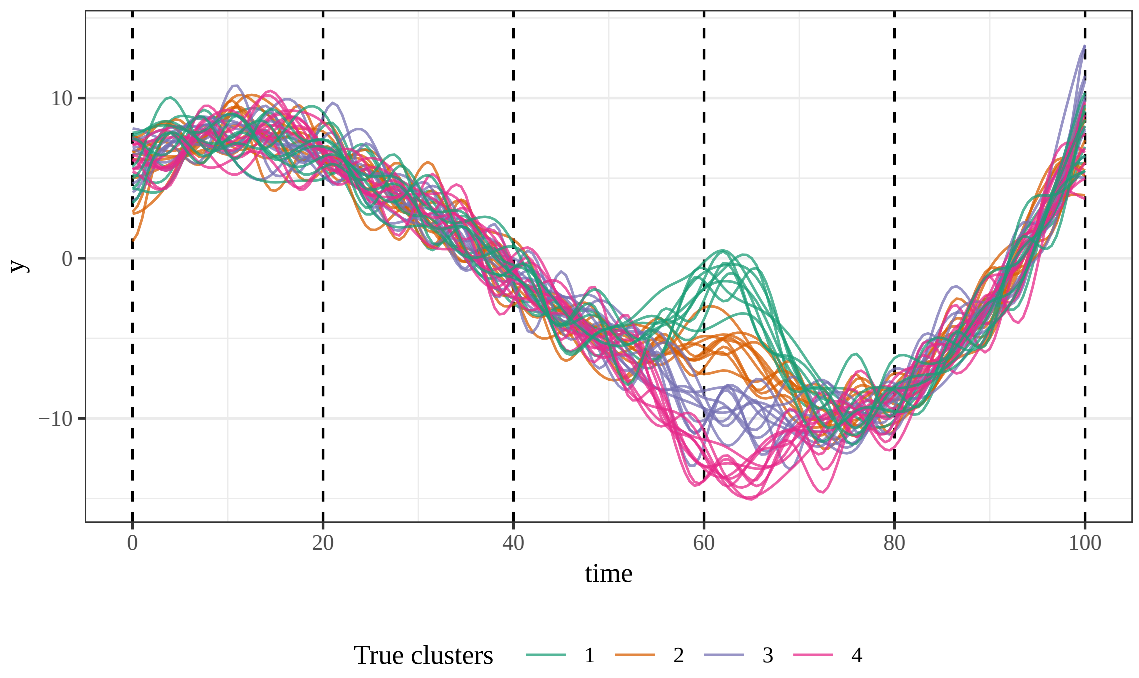


Figure 4.10: A smoothed simulated functional data set with four true clusters, colors based on true clusters.

Rand index) and in terms of variation explained (pseudo-R^2).

The Rand index (Rand, 1971) is a simple measure to compare two different cluster solutions. It counts the number of pairs of observations that either appear together or are separate in two clustering solutions (number of agreements), divided by the possible number of pairs from the data set. Therefore, a higher Rand index indicates a better agreement between two cluster solutions and 1 implies identical solutions. Hubert and Arabic (1985) argued that the original Rand index does not account for "the agreement by chance", which means that there is no common baseline for the "worst case" and hence it is difficult to say which cluster result is better than others based solely on the Rand index. They suggest an adjusted version of the index to correct that problem by subtracting both the numerator and denominator by the expected number of pairs in which the objects are placed in the same clusters in two

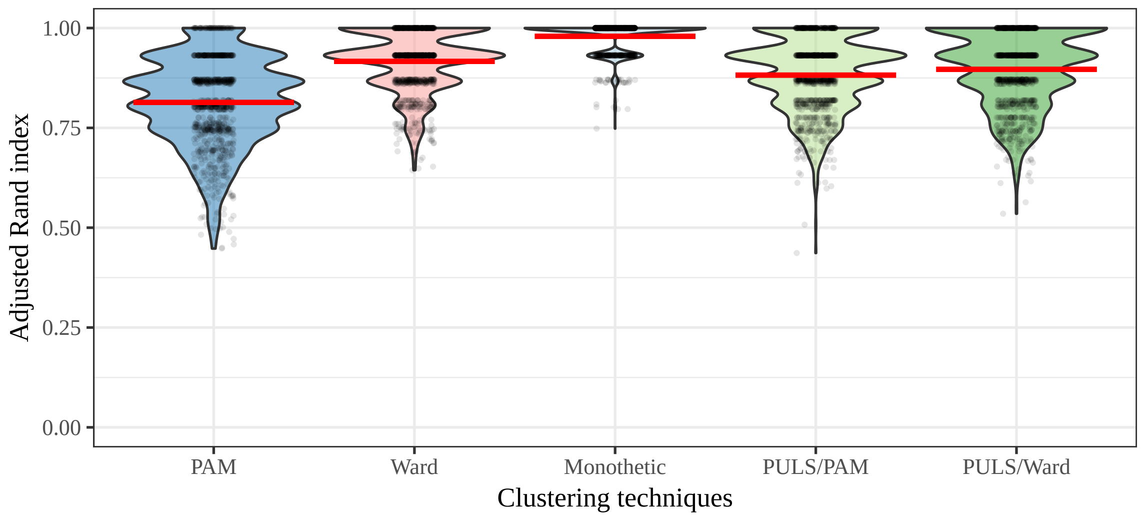


Figure 4.11: Adjusted Rand index of the five clustering methods with the mean indices in red.

clustering results, thus ensuring that the adjusted Rand index is bound between 0 and 1. The adjusted Rand indices of the five clustering techniques on 1000 simulated functional data sets are depicted in Figure 4.11. We can see that two versions of PULS and monothetic clustering perform better than PAM and are competitive with Ward's method. Also, Ward's method works well in this simulation, both in PULS and by itself.

We also evaluate the variation around the cluster mean of the generated data to see how well the clustering techniques do in this simulation study. The proportion of inertia (Eq. 4.4) explained in each cluster is an adequate measure of the variation around the cluster mean to compare the clustering techniques (Chavent et al., 2007). Because the interpretation of this measure is similar to the explained variance, R^2, in regression models, we call this measure *pseudo*-R^2 for short. This criterion has been subject to some recent criticisms based on the potential to be inflated by scaling or projecting the data (Loperfido and Tarpey, 2018). Even with its potential issues

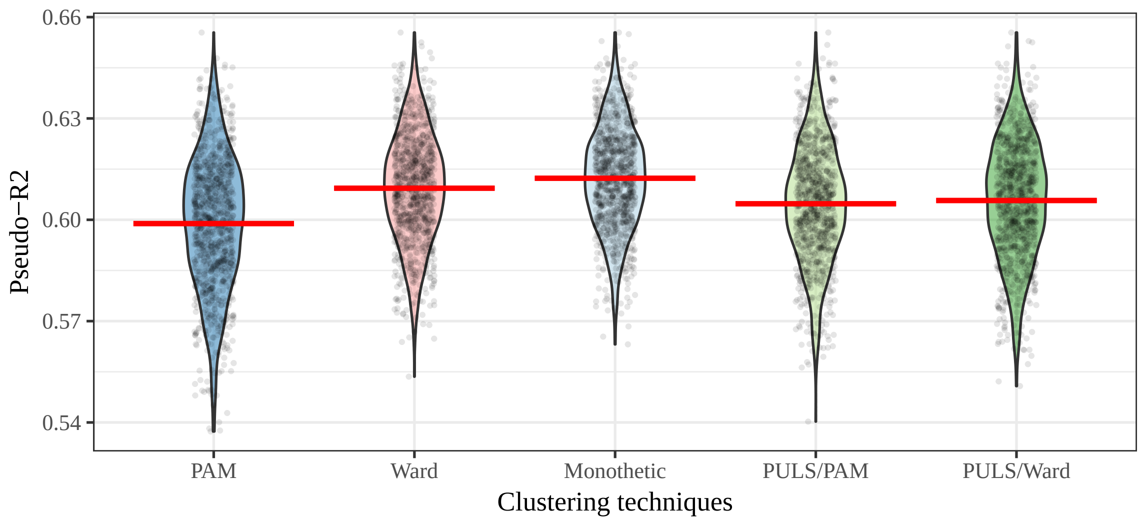


Figure 4.12: Pseudo-R2 of Clustering Methods. The y-axis limit is scaled to range from the minimum and maximum values of pseudo-R2.

from the transformations of data, it provides for interesting comparisons of different methods on the same distance matrix. The pseudo-R^2 for the five clustering methods on 1000 simulated sets of curves are shown in Figure 4.12. All five techniques seem to have competitive pseudo-R^2's in this simulation study. They all explain a moderate amount of variation around cluster means (55% to 65%) and are consistent partly because of the constant chosen standard deviation of the random noises. Although it appears there are some differences between methods based on the pseudo-R^2 criterion, with monothetic technique performed slightly better than others, the variations are high compared to the difference in mean pseudo-R^2 between techniques. Therefore, there is limited evidence of differences in terms of pseudo-R^2 criterion across the methods considered.

PULS had slightly lower levels of performance here but improved on the PAM algorithm and did not work much worse than Ward's method. Surprisingly, monothetic clustering of discretely evaluated functional data provided the best results.

4.5 Conclusions and Extensions

Cluster analysis is often used when researchers want to figure out the underlying structure of data where little or no knowledge of that structure or expectation of the structure to be validated exists. Using an appropriate clustering technique can result in many interesting insights to the data, as we have seen in the application to the Arctic sea ice extent data. Monothetic clustering has an advantage over other clustering algorithms in producing a set of splitting rules so the similarities and differences among clusters can be easily interpreted based on the defined splits. In particular, whether a year has low or high ice extent in January and July determined which cluster the year belonged to. Furthermore, it has potential to reduce the cost and effort by realizing which variables are influencing the separation of clusters so further work can focus on those variables specifically. The fact that the monothetic clustering constrained to only split using the summer months produced the same results when considering all days in a year for partitioning gives us the ability to predict the group a year should falls into by only examining the ice extent data in those summer months.

As no clustering technique fits all applications, monothetic clustering is not without its drawbacks. Functional data, as is the Arctic ice extent time series data, are considered as true continuous functions over an interval of an index (e.g., time, space, wavelength, or any other similar variables). Applying monothetic clustering, which requires discretely observed values over the index, to a function-based data set theoretically creates a problem of picking the variables to split when the difference in values between any two adjacent variables are very minor, especially when the functions are smooth across the measured index. That is the motivation for designing PULS, a new clustering technique inspired by monothetic clustering but based on splits defined in subregions of the index. PULS utilizes the properties of a functional data set, such as integrating the differences between all the values between two functions to calculate the L_2 distance matrix, gathering similar variables into “subregions” and exploiting the combined information to create partitions. From the simulation study, the choice of underlying clustering algorithm in PULS obviously affects its performance. Therefore, it is important to carefully consider the underlying technique suitable for a specific application. Also from the simulation study, monothetic clustering surprisingly provided good clustering performance in a situation that PULS should be more appropriate. Other simulation studies using more sparse discrete points to form the functional data may be better cases to show the ability of PULS to correctly identify clusters compared to monothetic clustering applied to discretized versions of functional data or the difference may just be in the interpretations of the clusters created by the two methods.

Cluster analysis of the Arctic ice extent over the years where data are available conforms with the conclusions of other research that, overall, the ice in the North Pole is decreasing in its extent (area). Other variables related to the Arctic ice such as the thickness of the ice could also be considered in the cluster analysis to give us another perspective on the changes of the ice in the area. We also learned that the ice extent information in summer months was associated with how that year’s trajectory would look. Because other factors were not considered in this study, we do not have other information to explain the trend in trajectories over time and there are other factors that may explain the complex changes of the ice extent patterns over time. Other research could explore inference techniques based on functional linear models to test for trends over time on the functional data we have generated (Ramsay and Silverman, 2005).

Also, deciding the adequate number of groups in the clustering of ice extent data should be examined further. In Chapter 2, we suggested some methods to address this issue in monothetic clustering. Last but not least, R packages (R Core Team, 2019) for monothetic clustering and PULS were developed to facilitate their further applications and visualizations in the future. A overview of those packages is in Chapter 5.

CHAPTER FIVE

MONOTHETIC CLUSTERING AND PARTITIONING USING LOCAL SUBREGIONS: THE R PACKAGES **MONOCLUST** AND **PULS**

Author: Tan V. Tran

Contributions: Responsible for the majority of the writing.

Co-Author: Mark C. Greenwood

Contributions: Provided guidance and feedback on statistical analysis and drafts of the manuscript.

Manuscript Information

Tan V. Tran and Mark C. Greenwood

The Journal of Open Source Software

Status of Manuscript:

__X__ Prepared for submission to a peer-reviewed journal

______ Officially submitted to a peer-reviewed journal

______ Accepted by a peer-reviewed journal

______ Published in a peer-reviewed journal

Abstract: Monothetic clustering is a clustering technique that has advantages in interpretability and prediction compared to other clustering techniques. It is based on recursive bipartitions of a data set by choosing the splitting rules using the variables, one at a time. Similar to other clustering techniques, monothetic clustering faces the problem of deciding the number of clusters. Permutation and cross-validation, two techniques that have been used in regression and classification tree-based methods, are adapted to use on monothetic clustering results to assist in this decision. Monothetic clustering, with modifications, has the potential to work on special types of variables, one of them involves data sets with circular variables. The unique characteristics of these variables pose challenges in performing the partitioning as well as visualization of the results. Measurements y_i that are taken over some ordered index t such as time, frequency, or space, but in principal, can be obtained as often as desired, are considered as functional data, $y(t)$. Functional data can also be partitioned by monothetic clustering if the functional data are evaluated on a common discrete set of times, frequencies, etc. Partitioning Using Local Subregions (PULS) is inspired by monothetic cluster analysis but is designed to better cluster functional data by defining splits using subregions, one at a time, in defining the recursive bipartitions. These two clustering techniques as well as visualizations and auxiliary functions to help researchers in making decisions related to clustering as implemented in the R *packages* ***monoClust*** *and* ***PULS*** *are discussed.*

5.1 Introduction

Cluster analysis (or clustering) attempts to group observations into clusters so that the observations within a cluster are similar to each other while different from those in other clusters. It is often used when dealing with the question of discovering structure in data where no known group labels exist or when there might be some question about whether the data contain groups that correspond to a measured grouping variable. Therefore, cluster analysis is considered a type of unsupervised learning. It is used in many fields including statistics, machine learning, and image analysis, to name just a few. For a general introduction to cluster analysis, see Everitt and Hothorn (2011, Chapter 6).

Commonly used clustering methods are k-means (MacQueen, 1967) and Ward's hierarchical clustering (Murtagh and Legendre, 2014; Ward, 1963), which are both implemented in functions `kmeans` and `hclust`, respectively, in the **stats** package in R (R Core Team, 2019). They belong to a group of methods called *polythetic clustering* (MacNaughton-Smith et al., 1964) which use combined information of variables to partition data and generate groups of observations that are similar on average. Monothetic cluster analysis (Chavent, 1998; Piccarreta and Billari, 2007; Sneath and Sokal, 1973), on the other hand, is a clustering algorithm that provides a hierarchical, recursive partitioning of multivariate responses based on binary decision rules that are built from individual response variables. It creates clusters that contain shared characteristics that are defined by these rules.

Given a clustering algorithm, the cluster analysis is heavily influenced by the choice of K, the number of clusters. If K is too small, it puts "different" observations together. On the other hand, if K is too large, the algorithm might split observations into different clusters that share many characteristics or are similar. Therefore, picking a sufficient, or "correct", K is critical for any clustering algorithm. A survey of some techniques for estimating the number of clusters has been done by Milligan and Cooper (1985). The R package **NbClust** (Charrad et al., 2014) is dedicated to implementations of those techniques. However, none of them were designed to work with monothetic clustering or take advantage of its unique characteristics, where binary splits generate rules for predicting new observations and each split is essentially a decision about whether to continue growing the tree or not. In Chapter 2, we examined M-fold cross-validation (a brief introduction can be seen in Hastie et al., 2016) and permutation-based hypothesis test at each split similar to those in Hothorn et al. (2006). These are two techniques that have been shown to work well in the classification and regression tree setting and we have adapted them to work with monothetic clustering.

Clustering data sets including circular variables, a type of variables measured in angles indicating the directions of an object or event (Fisher, 1993; Jammalamadaka and SenGupta, 2001) requires a different sets of statistical methods from conventional "linear" quantitative variables. An implementation of monothetic clustering modified to work on circular variables is discussed here. To assist in visualizing the resulting clusters and interpreting the shared features of clusters, a visualization of the results based on parallel coordinates plots (Inselberg and Dimsdale, 1987) are also implemented in the **monoClust** package. The package has been applied to the particle counts in föhn winds in Antarctica (`wind.sensit.bin.2007` and `wind.sensit.bin.2008` data sets from Šabacká et al., 2012) and the Arctic sea ice extent data (`Arctic_seaice` for data until December 31st, 2018 from Fetterer et al., 2018).

In an application of clustering to Arctic sea ice extent data comprising daily measurements from 1978 to present (Fetterer et al., 2018), we faced a challenge choosing one splitting variable among multiple equally qualified variables in monothetic clustering when applied to functional data. This happens because there are very similar observations in small intervals of time when the observations are smooth curves. A new clustering algorithm called Partitioning Using Local Subregions (PULS) that provides a method of clustering functional data using subregion information (described in detail in Chapter 4) is implemented in the R package **PULS**. It is designed to complement the **fda** and **fda.usc** packages (Febrero-Bande and Fuente, 2012; Ramsay et al., 2018) in clustering functional data objects.

5.2 Monothetic Clustering

Let y_{iq} be the i^{th} observation ($i = 1, \dots, n$, the number of observations or sample size) on variable q ($q = 1, \dots, Q$, the number of response variables) in a data set. In cluster analysis, Q variables are considered "response" variables and the interest is in exploring potential groups in these responses. Occasionally other information in the data set is withheld from clustering to be able to understand clusters found based on the Q variables used in clustering. Clustering algorithms then attempt to partition the n observations into mutually exclusive clusters $C_1, C_2, \dots, C_K$ in Ω where K is the number of clusters, so that the observations within a cluster are "close" to each other and "far away" from those in other clusters.

Inspired by regression trees (Breiman et al., 1984) and the **rpart** package (Therneau and Atkinson, 2018), the monothetic clustering algorithm searches for splits from each response variable that provide the best split of the multivariate responses in terms of a global criterion called inertia. To run successfully on a data set, the `MonoClust` function of the **monoClust** package only has one required argument, which is the data set name in the `toclust` option. By default, `MonoClust` will be performed by first calculating the squared Euclidean distance matrix between observations. The calculation of the distance matrix is very important to the algorithm because *inertia*, a within-cluster measure of variability when Euclidean distance is used, is calculated by

$$I(C_k) = \sum_{i \in C_k} d^2_{euc}(\mathbf{y_i}, \overline{y}_{C_k}), \tag{5.1}$$

where $\overline{y}_{C_k}$ is the mean of all observations in cluster C_k. This formula has been proved to be equivalent to the scaled sum of squared Euclidean distances among all observations in a cluster (James et al., 2013, p 388),

$$I(C_k) = \frac{1}{n_k} \sum_{(i,j)\in C_k, i>j} d^2_{euc}(\mathbf{y_i}, \mathbf{y_j}).$$

A binary split, $s(C_k)$, on a cluster C_k divides its observations into two smaller clusters C_{kL} and C_{kR}. The inertia decrease between before and after the partition is defined as

$$\Delta(s, C_k) = I(C_k) - I(C_{kL}) - I(C_{kR}),$$

and the best split, $s^*(C_k)$, is the split that maximizes this decrease in inertia,

$$s^*(C_k) = \arg\max_s \Delta(s, C_k).$$

The same algorithm is then recursively applied to each sub-partition, recording splitting rules on its way until it reaches the stopping criteria, which can be set in `MonoClust` by at least one of these arguments:

- `nclusters`: the pre-defined number of resulting clusters;
- `minsplit`: the minimum number of observations that must exist in a node in order for a split to be attempted (default is 5);
- `minbucket`: the minimum number of observations allowed in a terminal leaf (default is minsplit/3).

The options `minsplit` and `minbucket` were adapted from **rpart**. The monothetic clustering algorithm is outlined in more detail in Algorithm 5.1.

As a very simple example, monothetic clustering of the *ruspini* data set (Ruspini, 1970) available in the **cluster** package (Maechler et al., 2018) with 4 clusters can be performed as follows:

```
library(monoClust)
ruspini4c <- MonoClust(cluster::ruspini, nclusters = 4)
ruspini4c

n= 75


Node, N, Within Cluster Deviance, Proportion Deviance Explained,
      * denotes terminal node


1) root 75 240000 0
  2) y < 91 35  43000 0.63
    4) x < 47 20   3700 0.16 *
    5) x >= 47 15   1500 0.16 *
  3) y >= 91 40  46000 0.63
    6) x < 68.5 23   3200 0.16 *
    7) x >= 68.5 17   4600 0.16 *
```

The output (`print.MonoClust`) lists each split on one line together with the splitting rule as well as its inertia and is displayed with the hierarchical structure so the parent–child relationships between nodes can be easily seen. This function defines a `MonoClust` object to store the cluster solution with some useful components:

- `frame`: a partitioning tracking table in the form of a `data.frame`, similar to **rpart**'s object;
- `Membership`: a vector of numerical cluster identification (incremented by new nodes) that observations belong to; and

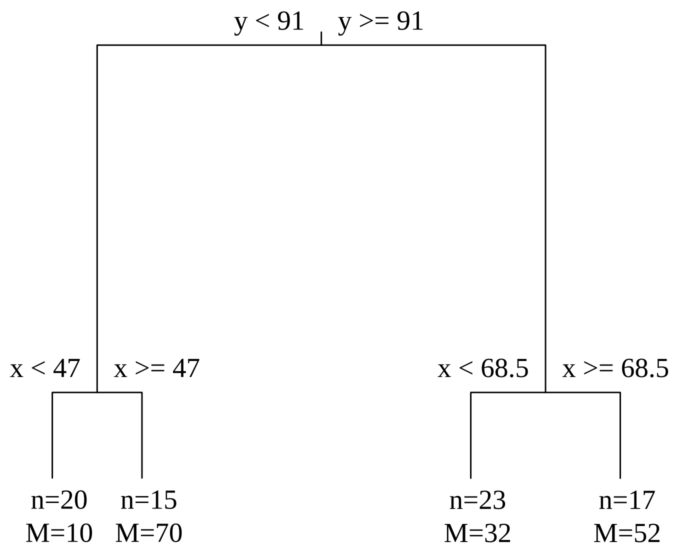


Figure 5.1: Binary partitioning tree with three splits, four clusters for *ruspini* data.

- `medoids`: the observation indices that are considered as representatives for the clusters (Kaufman and Rousseeuw, 1990), estimated as the observations that have minimum total distance to all observations in their clusters.

Another visualization of the clustering results is the splitting rule tree created by the `plot.MonoClust` function of the `MonoClust` object (Figure 5.1).

```
plot(ruspini4c)
```

The initial development of **MonoClust** was based on **rpart**. It borrows the contents of **rpart** in many places: arguments, text output, tree output, and the `MonoClust` object structure. However, we made many modifications to refine and enhance the structure to implement the various methods described in Chapters 2 and 3. We describe the modifications in the following sections.

5.2.1 Testing at Each Split to Decide the Number of Clusters

Deciding on the number of clusters to report and interpret is an important part of cluster analysis. Among many metrics mentioned in Milligan and Cooper (1985) and Hardy (1996), Caliński and Harabasz (CH)'s pseudo-F (Caliński and Harabasz, 1974) is among the metrics that have typically good or even the best performance in the Milligan and Cooper (1985)'s simulation studies on selecting the optimal number of clusters. Additionally, average silhouette width (AW), a measure of how "comfortable" observations are in their clusters they reside in, has also been suggested to select an appropriate number of clusters (Rousseeuw, 1987). One limitation of both criteria is that they are unable to select a single cluster solution because their formula require at least two clusters to calculate the criteria. We have proposed (in Chapter 2) two methods that can assist select the number of clusters in monothetic clustering. One is inspired by regression tree methods for pruning regression and classification trees, which is an adaption of M-fold cross-validation technique. Another one is inspired by conditional inference trees (Hothorn et al., 2006). It is a formal hypothesis test at a split to determine if it should be performed using two different test statistics. Finally, we suggested a hybrid method that uses the hypothesis test with CH's F statistic at the first split and then uses the original CH's F for the further splits if the test suggests that there should be at least two clusters.

The M-fold cross-validation randomly partitions data into M subsets with equal (or close to equal) sizes. $M - 1$ subsets are used as the training data set to create a tree with a desired number of leaves and the other subset is used as validation data set to evaluate the predictive performance of the trained tree. The process repeats for each subset as the validating set ($m = 1, \ldots, M$) and the mean squared difference,

$$MSE_m = \frac{1}{n_m} \sum_{q=1}^{Q} \sum_{i \in m} d^2_{euc}(y_{iq}, \hat{y}_{(-i)q}),$$

is calculated, where $\hat{y}_{(-i)q}$ is the cluster mean on the variable q of the cluster created by the training data where the observed value, y_{iq}, of the validation data set will fall into, and $d^2_{euc}(y_{iq}, \hat{y}_{(-i)q})$ is the squared Euclidean distance (dissimilarity) between two observations at variable q. This process is repeated for the M subsets of the data set and the average of these test errors is the cross-validation-based estimate of the mean squared error of predicting a new observation,

$$CV_K = \overline{MSE} = \frac{1}{M} \sum_{m=1}^{M} MSE_m.$$

The purpose of the cross-validation is to find a cluster solution that achieves the "best" prediction error for new observations. There are several ways one can decide from the output of `MonoClust`. A naive approach is to pick the solution that has the smallest CV_K (*minCV* rule). However, in many cases, it can result in a very high number of clusters if the error rate keeps decreasing even though there is often a small change after a few large drops. To avoid this problem, Breiman et al. (1984) suggested picking the solution that is simplest within 1 or 2 standard errors (SE) from the minimum error estimate (*CV1SE* or *CV2SE* rules), with the standard error is defined as

$$SE(\overline{MSE}) = \sqrt{\frac{1}{M} \sum_{m=1}^{M} (MSE_m - \overline{MSE})}.$$

The function `cv.test` with the data set and two arguments `minnodes` and `maxnodes` defining the range of nodes to test on will apply `MonoClust` and calculate both $\overline{MSE}$

(which is named MSE in the output) and its standard error (named Std. Dev. as used in **rpart** package).

```
cp.table <- cv.test(ruspini, minnodes = 1, maxnodes = 10)
cp.table

     MSE Std. Dev.
1  25263      6567
2   9942      2607
3   7770      2821
4   1811      1438
5   1666      1429
6   1606      1270
7   1510      1255
8   1479      1263
9   1424      1274
10  1362      1285
```

A plot with error bars for one standard error, similar to Figure 5.2, can be made from the output table using standard plotting functions to assist in assessing these results.

Another approach involves doing a formal hypothesis test at each split on the tree and using the p-values to decide on how many clusters should be used. This approach has been used in the context of conditional inference trees by Hothorn et al. (2006) although with a different test statistic and purpose. In that situation the test is used to test a null hypothesis of independence between the response and a selected predictor. For cluster analysis, at any cluster (or leaf on the decision tree), whether it will be partitioned or not is the result of a hypothesis test in which the pair of hypotheses can be abstractly stated as

H_0 : The two new clusters are identical to each other, and

H_A : The two new clusters are different from each other.

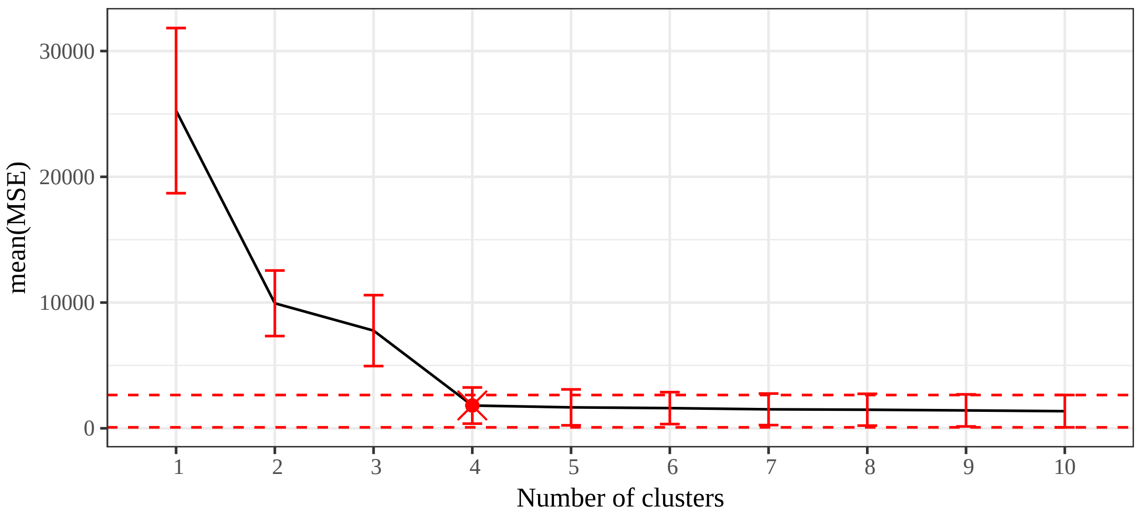


Figure 5.2: The choice of clusters for Ruspini data made by 10-fold CV where $minCV$ selects 10 clusters and *1SE* selects 4. The error bars are the $\overline{MSE} \pm 1SE$ and the choice of 4 clusters, the simplest solution within 1 standard error of the minimum error estimate (the dashed lines coincide with the bar at 10 clusters) is highlighted with a $\times$.

To allow applications with any dissimilarity measure, a nonparametric method based on permutation is used. Anderson (2001) developed a multivariate nonparametric testing approach called $perMANOVA$ that involves calculating the pseudo-F-ratio directly from any symmetric distance or dissimilarity matrix where the sum of squares are, in turn, calculated from the dissimilarities. The p-value can then be calculated by tracking the pseudo-F across permutations and comparing the results to the observed result and is available in the **vegan** package (Oksanen et al., 2019) in R. In Chapter 2, we considered two approaches for generating the permutation distribution under the previous null:

1. Shuffle the observations between two proposed clusters. The pseudo-F's calculated from the shuffles create the reference distribution to find the p-value. Because the splitting variable that was chosen is already the best in terms of reduction of inertia, that variable is withheld from the distance matrix used

in the permutation test. This method can be done with `method = 1` (default value) in the `perm.test` function.

2. Shuffle the values of the splitting variables while keeping other variables fixed to create a new data set, then the average silhouette width (Kaufman and Rousseeuw, 1990) is used as the measure of separation between the two new clusters and is calculated to create the reference distribution. Specifying `method = 2` in `perm.test` will run this method.

3. Similar to the previous method but pseudo-F (as in the first approach) is used as the test statistic instead of the average silhouette width. This approach corresponds to `method = 3`.

Applying the `perm.test` function to a `MonoClust` object will add permutation-based p-values to the output of both `print` and `plot`. Users can specify the number of permutations with the `rep =` argument. An example of applying the cluster shuffling approach to the *ruspini* data set follows and the tree output is in Figure 5.3. Note that the Bonferroni-adjusted p-values are used to account for the multiple hypothesis tests required when going deeper into the tree. The number of tests for the adjustment is based on the number of tests previously performed to get to a candidate split and the maximum value of a p-value is always 1. A similar adjustment has been used in conditional inference trees and was also implemented in its accompanied **party** package (Hothorn et al., 2006).

```r
ruspini6c <- MonoClust(ruspini, nclusters = 6)
ruspini6c.pvalue <- perm.test(ruspini6c, data = ruspini, method = 1, rep = 1000)
plot(ruspini6c.pvalue, branch = 1, uniform = TRUE)
```

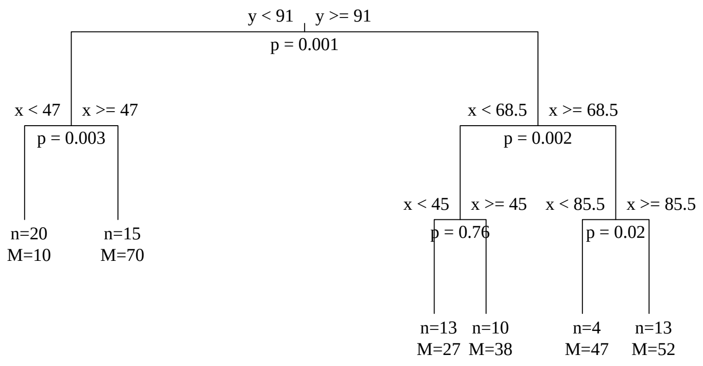


Figure 5.3: Binary partitioning tree with five splits, six clusters, but one split should be pruned based on its p-value of 0.8.

5.2.2 Clustering on Circular Data

In many applications, a variable can be measured in angles, indicating the directions of an object or event. Examples could be the times of day, aspects of the slope in mountainous terrain, directions of motion, or wind directions. Such variables are referred to as *circular variables* and are measured either in degrees or radians relative to a pre-chosen 0 degree position and meaning of a rotation direction. There are books dedicated to this topic (for example, Fisher, 1993; Jammalamadaka and SenGupta, 2001) that develop parametric models and analytic tools for circular variables. Here we demonstrate multivariate data analysis involving circular variables, such as visualization and clustering, as discussed in Chapter 3.

Cluster analysis depends on the choice of distance or dissimilarity between multivariate data points. A (dis)similarity measure that often comes up in the literature when dealing with mixed data types is Gower's distance (Gower, 1971). It is a similarity measure among observations from various types of variables, such as quantitative, categorical, and binary, can be a reasonable alternative to Eq. 5.1 when working with "mixed" data.

Generally, the Gower's dissimilarity in a simple form (without weights) for a data set with Q variables is

$$d_{gow}(\mathbf{y_i}, \mathbf{y_j}) = \frac{1}{Q}\sum_{q=1}^{Q} d_{gow}(y_{iq}, y_{jq}).$$

If q is a linear quantitative variable,

$$d(y_{iq}, y_{jq}) = \frac{|y_{iq} - y_{jq}|}{\max_{i,j}|y_{iq} - y_{jq}|}.$$

It can also incorporate categorical variables, with $d(y_{iq}, y_{jq})$ equals to 0 if the two observations belong to the same category of q and 1 otherwise. Details and examples can be seen in Everitt et al. (2011, Chapter 3). We extend the dissimilarity measure for a circular variable as

$$d(y_{iq}, y_{jq}) = \frac{180 - |180 - |y_{iq} - y_{jq}||}{180},$$

where α and β are the angles in degree. If radians are used, the constant 180 degrees will be replaced by π. This distance can be mixed with other Gower's distances both for monothetic clustering and in other distance-based clustering algorithms.

We demonstrated an application of monothetic clustering to a data set from Šabacká et al. (2012). This data set is a part of a study on microorganisms carried in föhn winds at the Taylor Valley, an ice free area in the Antarctic continent. The examined subset of the data is during July 7–14, 2008, at the Bonney Riegel location with three variables: the existence of particles measured in 1 minute every 15 minutes (binary variable of presence or absence), average wind speed (m/s), and

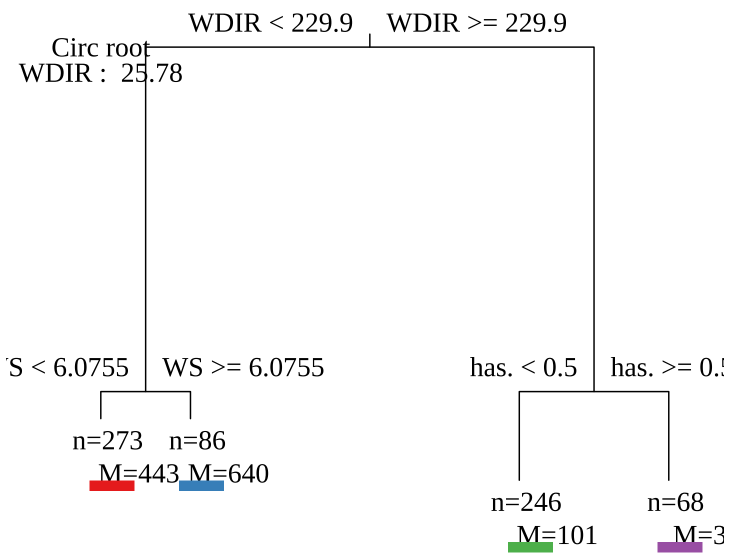


Figure 5.4: Splitting rule for the four-cluster solution. The color at the node can be set by `cols` argument. They match the ones in Figure 5.5.

wind direction (degrees) recorded at a nearby meteorological station every 15 minutes. Wind direction is a circular variable in which winds blowing from the north to the south were chosen to be 0/360 degrees and winds blowing from the east to the west were chosen to be 90 degrees. `MonoClust` works on circular data by indicating the index or name of the circular variable (if there is more than one circular variable, a vector of them can be transferred) in the `cir.var` argument.

```r
sensit042008 <- MonoClust(wind.subset.bin.2008, nclusters = 4, cir.var = 3)
```

To perform monothetic clustering, a variable must generate a binary split. Because of special circular characteristics, a circle needs two cuts to create two separate arcs instead of one cut-point as in conventional linear variables. Therefore, the algorithm to search for the best split in a circular variable is actually done in two folds; first by fixing one cut value and then searching for the second cut. This process is repeated by changing the first cut until all possible pairs of cuts have been examined and the best two cuts are then picked based on the inertia. The splitting rule tree is also updated to add the second split value on the corner of the tree. After the first split on a circular variable, the arcs can be considered as two conventional quantitative variables and can be split further with only a single cut-point. Figure 5.4 shows the resulting four clusters created by applying monothetic clustering on the Antarctic data.

When clustering a data set that has at least one circular variable in it, visualizing the cluster results to detect the underlying characteristics of the clusters is very crucial. Scatterplots are not very helpful for circular data because of the characteristics of those variables. Dimension reduction can be performed using techniques like multi-dimensional scaling (Chapter 14, Hastie et al., 2016) or more recent techniques such as t-SNE (Maaten and Hinton, 2008), but the details of the original variables are lost in these projections. Parallel Coordinates Plots (PCPs, Inselberg and Dimsdale, 1987), which can display the original values of all of the multivariate data by putting them on equally spaced vertical x-axes, are a good choice due to its simplicity and its capability to retain the proximity of the data points (Härdle and Simar, 2015, Chapter 1). A modified PCP, inspired by Will (2016), is also implemented in **monoClust** using **ggplot2** (Wickham, 2016). The circular variable is displayed as an ellipse with the options to rotate and/or change the order of appearance of variables to help facilitate the detection of underlying properties of the clusters. Figure 5.5 is the PCP of the Antarctic data with the cluster memberships colored and matched to the tree in Figure 5.4 with the following code. There are other

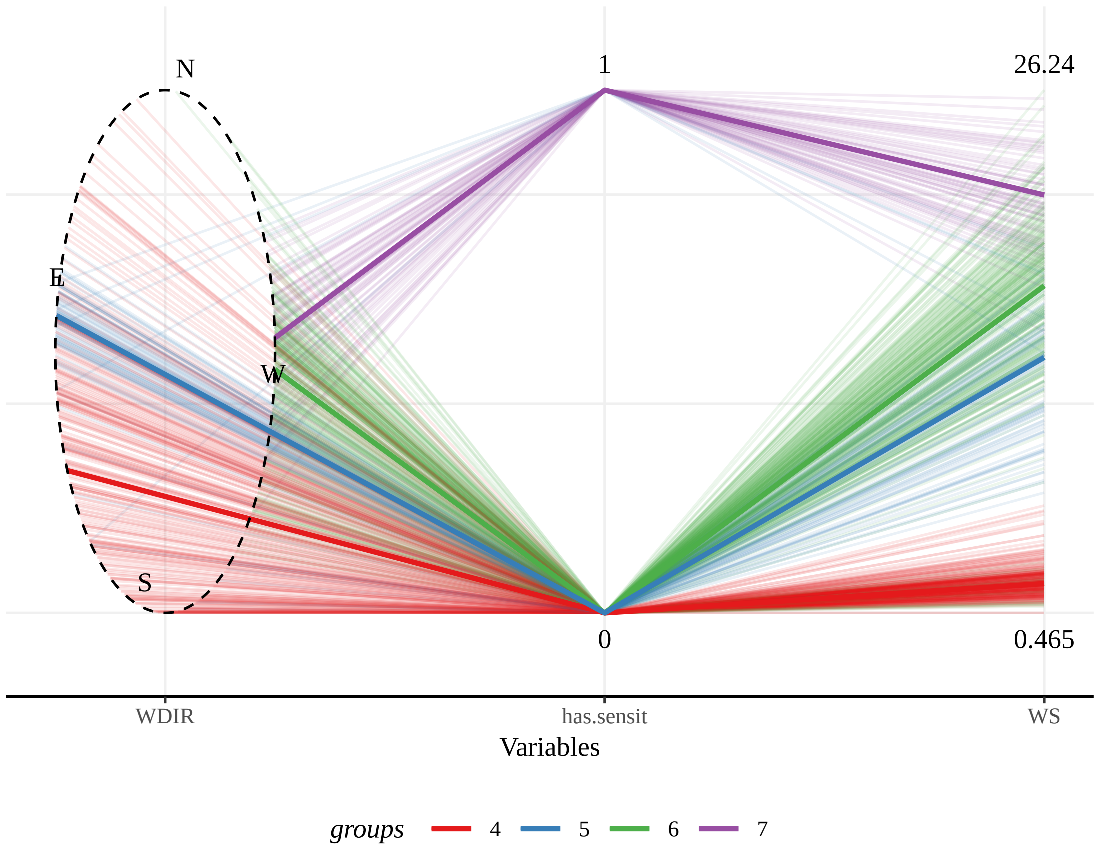


Figure 5.5: PCP with the circular variable (*WDIR*) depicted as an ellipse. The geographical direction is noted and the ellipse is rotated to facilitate understanding of clusters.

display options that can be modified such as the transparency of lines, whether the circular variable is in degrees or radians, etc. (see the function documentation for details).

```
pcp.gg(data = wind.sensit.bin.2008, circ.var = "WDIR",
       order.appeaer = c("WDIR", "has.sensit", "WS"),
       cluster.sol = sol42008,
       cols = c("#e41a1c", "#377eb8", "#4daf4a", "#984ea3"),
       shift = pi/4+0.6)
```

5.3 Partitioning Using Local Subregions (PULS)

Measurements y taken over some ordered index such as time, frequency, or space and thought of as curves or functions of the index t and/or other predictors are called *functional data* and denoted as $y(t)$ (Ramsay and Silverman, 2005). Functional data have, possibly, a high frequency of observations over the index t and are assumed to be generated by a smooth underlying process. Some examples of data that can be treated as functional include the growth curves for girls in the Berkeley Growth Study (Tuddenham and Snyder, 1954), hydraulic gradients in wetlands (Greenwood et al., 2011), or daily ice extent over years in the Arctic Sea (Fetterer et al., 2018). Clustering can be useful for functional data to find groups of curves sharing common characteristics and to find representative curves corresponding to different modes of variation in the data.

Functional clustering requires (1) construction of a functional data object and (2) application of a clustering algorithm either to the functional data directly or to a distance matrix calculated from the functional data. In R, Ramsay et al. (2018) created a package named **fda** to do the former task and function `metric.lp` in **fda.usc** (Febrero-Bande and Fuente, 2012) is designed to find the L_2 distance matrix for functional data in the latter approach. The code for creating an `fda` object is shown in the Appendix because they are not the focus of this paper.

In some functional data applications, there is pre-existing knowledge of regions of interest such as intervals of time where the curves are expected to be very different from each other. Partitioning using local subregions (PULS) is a clustering technique designed to explore subregions of functional data for information to split the curves into clusters. After defining the subregions $[a_1, b_1], [a_2, b_2], \dots, [a_R, b_R]$, the Euclidean ($L_2$) distance is calculated between functions $y_i(t)$ and $y_j(t)$ (Febrero-Bande and

Fuente, 2012) using the function `metric.lp` in **fda.usc** to provide

$$d_R(y_i, y_j) = \sqrt{\int_{a_r}^{b_r} [y_i(t) y_j(t)]^2 dt}$$

and obtain a dissimilarity matrix for each subregion, $r = 1, \ldots, R$. Adapting the idea of the monothetic clustering algorithm, each subregion is separately considered as a splitting candidate for the next 2-group partitioning, using commonly used clustering techniques such as k-means (MacQueen, 1967), Ward's method (Ward, 1963), or partitioning around medoids (PAM, Kaufman and Rousseeuw, 1990). An inertia-like criterion is again used as the global criterion for selecting each split. Among the K candidate splits, one from each subregion, the one having the largest decrease in inertia will be chosen as the best split. The algorithm is then recursively applied to the resulting sub-partitions until a specified number of partitions is reached or a stopping rule is met. Pseudocode for this algorithm is in Algorithm 5.2.

The PULS algorithm was inspired by monothetic clustering and shares many features such as inertia as the global criterion, a recursive bi-partitioning method, and the same stopping rules. However, the idea and applications of PULS are different enough to store in a separate R package, which we named **PULS**. Indeed, **PULS** borrows many private functions from **monoClust** such as the splitting rule tree, recursively checking for the best splitting subregion, tree-based displays of results, etc. An example of a usage of the main function of the package, `PULS`, to the Arctic ice extent data with subregions defined by months follows.

```
library(PULS)


Jan <- c(1, 31); Feb <- c(31,59); Mar <- c(59,90); Apr <- c(90,120);
May <- c(120,151); Jun <- c(151,181); Jul <- c(181,212); Aug <- c(212,243);
```

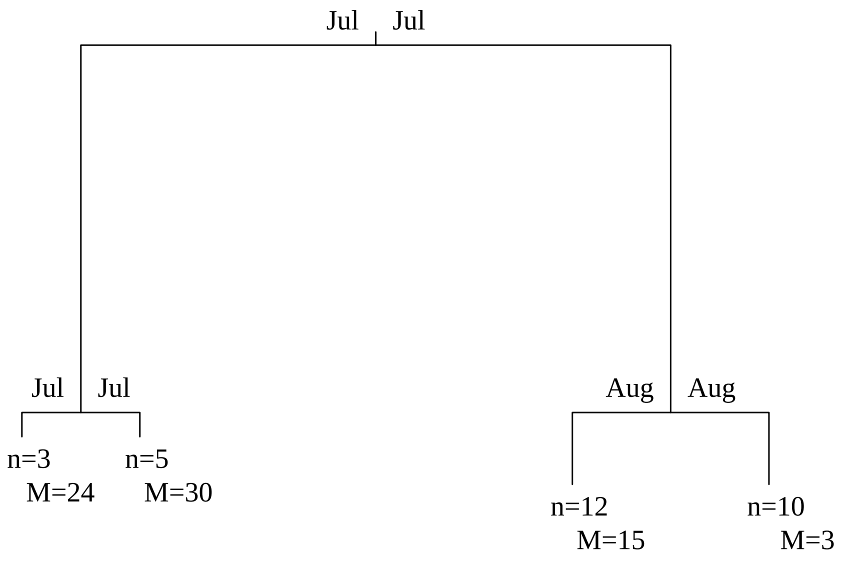


Figure 5.6: Four cluster result of PULS on Arctic ice extent data for years 1979–1986, 1989–2013.

```
Sep <- c(243,273); Oct <- c(273,304); Nov <- c(304,334); Dec <- c(334,365)


intervals <- rbind(Jan, Feb, Mar, Apr, May, Jun, Jul, Aug, Sep, Oct, Nov, Dec)
PULS4.pam <- PULS(toclust.fd=yfd.train$fd, intervals=intervals, nclusters=4,
                      method = "pam")
plot(PULS4.pam)
```

Instead of having data set name as a required argument in **monoClust**, the required arguments in `PULS` are an `fda` object created by applying the `smooth.basis` function (see Appendix for the codes) and the list in a `data.frame` of specified intervals for the subregions of t. Other arguments in `PULS` function include:

- `method`: the clustering method used to partition the subregions. It can be chosen between `"pam"` (for PAM method) and `"ward"` (for Ward's method).
- `distmethod`: the method for calculating the distance matrix. It can be either `"usc"` for the `metric.lp` in **fda.usc** package (default) or `"manual"` for using the inner product of a very fine grid of values between two functions.
- `labels`: the name of entities
- `nclusters`: the desired number of clusters in the results
- `minbucket` and `minsplit`: the minimum number of observations that must exist in a node in order for a split to be attempted and the minimum number of observations allowed in a terminal leaf, respectively.

5.4 Constrained Versions of the Two Clustering Techniques

In a data set, not all variables are created equal. Some variables are easier or cheaper to measure than others. For example, a data set on human vital information may include body fat percentage, height, and weight. Height and weight are much easier to obtain accurate measurements for than body fat percentage. If the data can be partitioned using only these easy-to-measure variables and the results are comparable to using all variables, it can help reduce effort and cost in the future studies where new subjects could be placed in clusters using easy or cheap to measure variables only.

Both monothetic clustering and the PULS algorithm implemented in the **monoClust** and **PULS** packages have options to limit the splitting candidates to a subset of variables (in case of monothetic) or subregions (in case of PULS) by specifying the argument `variables =`. The following code limits the subset of splitting variables only to summer months (August and September) of the Arctic ice extent data. Besides the benefits of reducing effort and cost in future studies, the constrained versions of the algorithms also speed up the running time, making the cluster estimation faster.

In **monoClust** and **PULS**, the constrained sets of splitting candidates can be set in `variables` and `spliton` arguments, respectively. The example code below runs `MonoClust` and `PULS` allowing only partitions based on a subset of days from day 181 to 273 (or July to September in the case of PULS).

```r
constrained.mono <- MonoClust(arctic, nclusters=4, variables = 181:273)
constrained.PULS <- PULS(yfd.train$fd, intervals = intervals, nclusters=4,
                         spliton = 7:9, method = "pam")
```

5.5 Conclusions

While exploring various problems related to monothetic clustering, we created and expanded the **monoClust** package to facilitate our study. It not only applies the monothetic clustering, a clustering technique that has interpretation advantages, to various kinds of data such as data with functional or circular variables, but also allows visualizing the cluster results to further enhance the interpretation of the results. It also has functionality to help users choose a reasonable number of clusters with cross-validation and permutation-based methods. Furthermore, **PULS** can be considered as a sister project of monothetic clustering using many core functions of **monoClust** but modified to work on functional data and data with pre-defined local subregions. They complement each other to explore interesting and useful applications for these techniques.

There is always room for improvements of both packages. Currently, efficiently accounting for data with many-category categorical variables and mixes of these with other variables are being considered. However, sorting and searching for the best splits in categorical variables are still non-trivial problems. The run time could also be improved by optimizing the search algorithms such as utilizing a local maximum search, both in "linear" and circular versions, such as using the monkey search (Chen et al., 2014) or using random walks (Russell et al., 2010).

Further documents, script files, and test versions of these packages are available at `https://github.com/vinhtantran/monoClust` and `https://github.com/vinhtantran/PULS`.

Appendix to Chapter Five: Creating an FDA Object in R

```r
library(fda)

# Step 1: using GCV to select the optimal lambda for each curve (year)

NORDER <- 4

gcv <- function(lambda, lapse, i) {

  ydata <- lapse %>%

    filter(Year == i) %>%

    select(yday, Extent)


  splinebasis <- create.bspline.basis(rangeval = c(min(ydata$yday),

                                                    max(ydata$yday)),

                                      breaks = ydata$yday,

                                      norder = NORDER)

  fdParobj <- fdPar(fdobj = splinebasis,

                    Lfdobj = 2,

                    lambda = lambda)

  ice.fd.obj <- smooth.basis(argvals = ydata$yday,

                             y = ydata$Extent,

                             fdParobj = fdParobj)

  return(ice.fd.obj$gcv)

}


year.lst <- unique(north.extra[["Year"]])

lambda.lst <- vector("list", length(year.lst))


for (i in seq_along(year.lst)) {

  upper <- 1

  lambda <- optim(par=0.2, gcv, lower = 0.001,

                  upper = upper, lapse = north.extra, i = year.lst[i])
  print(upper)
  while (lambda$par == upper) {
    upper <- upper + 5
    lambda <- optim(par=0.2, gcv, lower = lambda$par,
                    upper = upper, lapse = north.extra, i = year.lst[i])
    print(upper)
  }
  lambda.lst[[i]] <- lambda$par
}

lambda.lst <- unlist(lambda.lst)

# Step 2: creating the B-spline basis based on the found lambda values, and
# create an FDA object from the basis functions, then predict the missing values
# based on the functional data. This step is neccessary for applying monothetic
# clustering.
NORDER <- 4

year.lst <- unique(north2017.extra[["Year"]])

predicted.mat <- matrix(NA, 37, 366)

for (i in 1:length(year.lst)) {
  ydata <- north2017.extra %>%
    filter(Year == year.lst[i]) %>%
    dplyr::select(yday, Extent)

  splinebasis <- create.bspline.basis(rangeval = c(min(ydata$yday),
                                                   max(ydata$yday)),
                                      breaks = ydata$yday,
                                      norder = NORDER)
  fdParobj <- fdPar(fdobj = splinebasis,
                    Lfdobj = 2,
                    lambda = lambda.lst[i])
  icefd <- smooth.basis(argvals = ydata$yday,
                             y = ydata$Extent,
                             fdParobj = fdParobj)$fd
  predicted.mat[i,] <- predict(icefd, newdata = 1:366)
}

# Step 3: Re-create the FDA object with no lambda any more. This step is not
# really neccessary if applying PULS.
y<-t(predicted.mat)
splinebasis <- create.bspline.basis(rangeval=c(1, 366),
                                    nbasis=NBASIS,norder=NORDER)
fdParobj<-fdPar(fdobj=splinebasis,Lfdobj=2,lambda=.000001)
yfd.full<-smooth.basis(argvals=1:366, y=y, fdParobj=fdParobj)

plot(yfd.full)

yfd.train<-smooth.basis(argvals=1:366,
                        y=t(predicted.mat[1:(nrow(predicted.mat)-3), ]),
                        fdParobj=fdParobj)
```

Algorithm 5.1: Monothetic clustering

```
function FINDSPLIT(data):
    for all variable in the data set do
        Sort the data set by the variable in increasing order
        for all value in the variable do
                Bi-partition the data set at the value
                Calculate the inertia of newly created clusters $I(g_L)$ and $I(g_R)$
                Calculate the decrease in inertia $I(g) - I(g_L) - I(g_R)$
        end for
        Store the value that maximizes the change in inertia
    end for
    Choose the variable that creates the maximum change in inertia
function PARTITION:
    Start with one cluster of all observations
    repeat
        for all cluster k do
                FINDSPLIT(k)
        end for
        Bi-partition the data set at the best split
    until the number of desired clusters has been reached OR the minimum cluster
size has been reached
```

Algorithm 5.2: Partitioning using local subregions

```
function FINDSPLIT(data):
    for all subregion do
        Bi-partition the data set by a clustering method
        Calculate the inertia of newly created clusters $I(g_L)$ and $I(g_R)$
        Calculate the decrease in inertia $I(g) - I(g_L) - I(g_R)$
    end for
    Pick the subregion whose split creates the maximum change in inertia
function PARTITION:
    Create Subregions, start with one cluster of all observations
    repeat
        for all cluster k do
            FINDSPLIT(k)
            Bi-partition the data set at the best split
        end for
    until the number of desired clusters has been reached OR the minimum cluster size has been reached
```

CHAPTER SIX

CONCLUSIONS AND POSSIBLE EXTENSIONS

Cluster analysis is a widely used method to detect underlying structure when no information about that structure may be known. In this dissertation, we focus on monothetic clustering, a type of clustering technique in which the data are bi-partitioned at the values of the variables, one at a time. Therefore, the clusters would share the same characteristics, such as the same interval of a quantitative variable, the same category of a categorical variable, or the same directions of a directional variable. This feature of monothetic clustering gives the users the ability to interpret the clusters and predict the cluster a new observation would fall into. Selecting the number of clusters is a necessary step that any researcher using cluster analysis for optimization problems must do. Besides subjective methods such as picking the number of clusters that gives a desired result, looking for a sudden change in the plot of a clustering criterion against the number of clusters, or choosing the structure where there is clear separation in a dimension-reduced scatterplot (by using multi-dimensional scaling or principal component analysis), several objective methods are available to try to pick a reasonable number of clusters. Based on the features of the monothetic clustering algorithm suggested by Chavent (1998) and its tree of partitioning rules, we attempted to introduce criteria to automate the process of picking a reasonable number of groups in monothetic clustering. They are a cross-validated criterion inspired by CART (Breiman et al., 1984) and a permutation-based hypothesis test at each candidate split at tree nodes developed in the context of conditional inference trees (Hothorn et al., 2006). Various modifications, as well as novel suggestions to adapt these criteria to monothetic clustering, have been examined.

Via simulation studies, we found that the average silhouette width (AW) and Caliński and Harabasz (CH)'s pseudo-F, classic measures of fitness of clustering, are very good at detecting the true number of clusters of simulated data when using monothetic clustering when clear cluster structure is present. But these methods could not segregate between one and two cluster solutions. Meanwhile, the hypothesis test with the null distribution created from AW or CH's F by permuting the splitting variable and re-optimizing the shuffled data using the same variable was shown to be liberal and tends to choose a larger number of clusters than the truth. But the false null hypothesis rejection rate in the case of one true cluster was only slightly larger than the chosen significance value. Based on that result, we suggested a hybrid approach where a variable shuffling hypothesis test is used first to decide the clusterability of the data set. If there is evidence of more than one cluster, then we suggest using CH's pseudo-F to pick the number of clusters. We believe that this hybrid approach can be extended to other sequentially-defined (hierarchical) clustering algorithms. Specifically, shuffling values within the variables can break the correlation structure between variables and then re-clustering could be used to make the null distribution to test whether the first split (or last split) in divisive (agglomerative) clustering should be performed or not. After that, any standard criterion can be used to decide further splits/fuses if the testing method continues to be liberal in these applications.

The monothetic clustering algorithm explored here works with data with mixed categorical and quantitative variables (Chavent et al., 2007). In this dissertation, we explored the applications of monothetic clustering to other types of data, including data with circular variables and functional data. We extended Gower's dissimilarity measure and the splitting algorithm to work with circular variables. Thanks to the alternative formula to calculate the cluster inertia using any dissimilarity matrix (Equation 3.6), the monothetic clustering algorithm can work on versions of Gower's dissimilarity measure for mixed data types including a distance between angles in circular variables. We applied the method to data recording the particle counts carried by winds at the Taylor Valley in the Antarctic continent and the average speed and direction of the winds at 15-minute intervals during August 4–8, 2007 and July 7–14, 2008 at the Bonney Riegel location. In that data set, wind direction is a circular variable. Based on the resulting tree of partitioning rules, we could determine the separation of winds' features when a variable is used to partition the data and the association between different winds' variables when they together define the characteristics of a cluster. Specifically, winds were split into the down-valley föhn wind and the up-valley sea breezes. Most of the winds that fall into strong wind clusters flow in the föhn wind direction and most of the winds that fall into slow wind clusters flow in the opposite direction. Apart from the order of the splitting variables in the tree that can show the most influential variables on separating the data, further interpretations are not possible without the use of other visualizations associated with the clustering. Rose plots and a modified version of parallel coordinates plots to include a circular variable have been shown to be an effective way to visualize the cluster results. With some tweaks to the order of the variables and the rotation of the circles (shown as ellipses in the plot) implemented in the `pcp.gg` function in the **monoClust** R package (R Core Team, 2019), underlying features in the data, including the clusters, can be explored.

Another application of monothetic clustering is to functional data. Functional data analysis involves various techniques applied to data where measurements were taken over some ordered index such as time or space and are thought of as functions of that index. Like other types of data, finding structure in functional data using cluster analysis has been the interest of many researchers. An example data set of the daily ice extent area in the Arctic, which is freely available online (Fetterer et al., 2018), was examined to demonstrate the clustering of functional data. In the context of a functional data set, the ice extents in each year were used to estimate a smooth curve. There is a curve for daily ice extent across each year and each is a functional observation in the data set; day is the index of the functions. To apply monothetic clustering to functional data, the functional data are used in its discretized form where the functions, after being smoothed, are interpolated at a high resolution grid of values of the index, here day. This technique worked but suffered from the problem of deciding the cutoff values among many values that have the same amount of maximum decrease in inertia, which happened frequently because of the smooth nature of the functions. It also did not utilize the estimated functional data in their actual functional forms. Another partitioning algorithm called partitioning using local subregions (PULS), which was originally introduced by Dr. Mark Greenwood, was implemented to run on this data set. This clustering algorithm exploits the previous knowledge of information-shared subregions (sets of variables, or days in a month in the motivating data set) and considers all curves as continuous functions in the subregion to calculate the functional L_2 distance matrix among them. A clustering technique that uses a distance matrix as the input, such as Ward's method or partitioning around methods (PAM), is used to bi-partition the subregion. A global criterion like the cluster inertia used in monothetic clustering is used to decide which subregion would create the optimal split. This algorithm directly uses estimated functions to calculate the distance matrix. By using the combined information of curve segments in a subregion, it reduces the problem of having to choose among many equally optimal partitions. In the simulation study set up to evaluate the performance of the clustering algorithms on functional data where the true separation was designed to happen in a specific subregion, all the tested methods are very competitive. Monothetic clustering on the discretized functional data stood out with the highest adjusted Rand index and pseudo-R^2 criteria. PULS using Ward's method performed slightly better than using AW and was comparable to the original Ward's method.

Functional data usually have the number of variables (indices) much larger than the number of observations (functions). When used discretely, as in the way we used it to apply monothetic clustering, functional data have $n \ll Q$. Sparse clustering (Witten and Tibshirani, 2010) is a clustering method where a subset of features are used to show the most clear structure. It can be extended to functional data to potentially reduce the number of discrete points on the index to only those that are related to clearly defined clusters. Sparse clustering has two algorithm versions that are slightly different depending on whether the input of a clustering algorithm is the raw data set (as in k-means) or a dissimilarity matrix (as in hierarchical clustering). Essentially, the sparse clustering is an iterative process where a clustering technique is applied on weighted variables (or weighted distances in a dissimilarity matrix) with a fixed weight, then the resulting clustering structure is fixed to optimize the weights on the individual variables, where some can get zero weight. Extending the ideas of sparse clustering into functional data could be done discretely using weights at each value of the index going into the functional dissimilarity matrix or by estimating a weight function similar to the one used in Greenwood (2004) that could smoothly up or down-weight areas of the functional data. Sparse clustering ideas could also be applied to monothetic clustering in general or in the application to functional data as the weight function could be used to change the relative weights of components going into the dissimilarity matrix.

PULS could be helpful in other types of biological data sets such as grouping protein concentrations in multiple sclerosis (MS) patients and healthy subjects based on the biomarkers, grouped by the cellular origin and known function of individual components (Barbour et al., 2017). The clustering of this data set may shed light on what groups of biomarkers are associated with the separation of healthy subjects and MS patients.

We could also revisit the data sets explored in the previous chapters in a different way. An interesting example would be the clustering of daily Arctic ice extent data. We have assessed the functional nature of the nearly continuous ice extent in the area but we have not tackled on the periodic nature of days/years besides using that information to make sure the smoothed curves are uninterrupted at the December 31 and January 1 cut points. Days of year can be considered circular; days can be displayed on a circle with day 365/366 and day 1 adjacent. By doing that, the resulting clusters would better reflect the time intervals they contain when using monothetic clustering. The ice extent data can be displayed as one curve spiraling around the circle and the cluster structure can be spotted without discontinuity artificially imposed beginning and end of years (Figure 6.1).

The ability to constrain the set of candidate splitting variables of the monothetic and PULS algorithms implemented in the **monoClust** and **PULS** packages has potential to assist researchers in realizing the importance of a subset of the measured variables in segregating the data portions with shared characteristics. A cluster result of a clustering method constrained on a set of easy-to-measure variables, if it is very similar to that of the unconstrained clustering method, will help the researchers focus on those easy-to-measure variables in the future studies on similar goals to reduce the effort and cost spent to collect the data.

To test the methods and evaluate the methods used in this dissertation, we developed and maintained two R packages hosted at Github. When implementing the

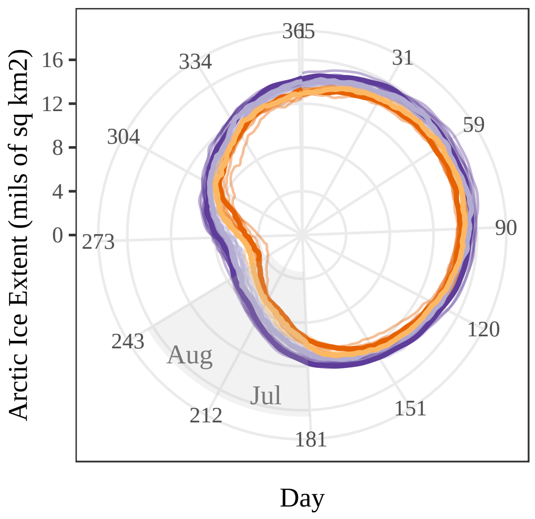


Figure 6.1: The Arctic ice extent data displayed as one curve spiraling around a yearly circle. The colors are clusters resulting from four cluster solution in PULS (in Chapter 4).

monothetic clustering algorithm proposed by Chavent (1998), we had to make many design and implementation choices in R to make the algorithms work efficiently. Most of the computational challenge in monothetic clustering is to find the optimal split across all variables. The current version of the algorithm exhaustively searches for all possible splits within a variable, then repeats the process for all variables to pick a split that is guaranteed to be globally optimal. The running time was acceptable (on our personal computer) running on reasonable sized data sets (such as the ice extent data set with $n = 40 \times Q = 365$) but would take 10 minutes to run just on a subset of the overall wind particle counts data ($n = 679 \times Q = 3$ with one circular variable). This problem is multiplied when simulation studies are considered. The speed of the search process also can be improved by using other heuristic optimal search algorithms instead of the current brute force approach. However we would need to consider that the optimization function might be multimodal and the search domain consists of discrete values. Figure 6.2 shows these characteristics by plotting

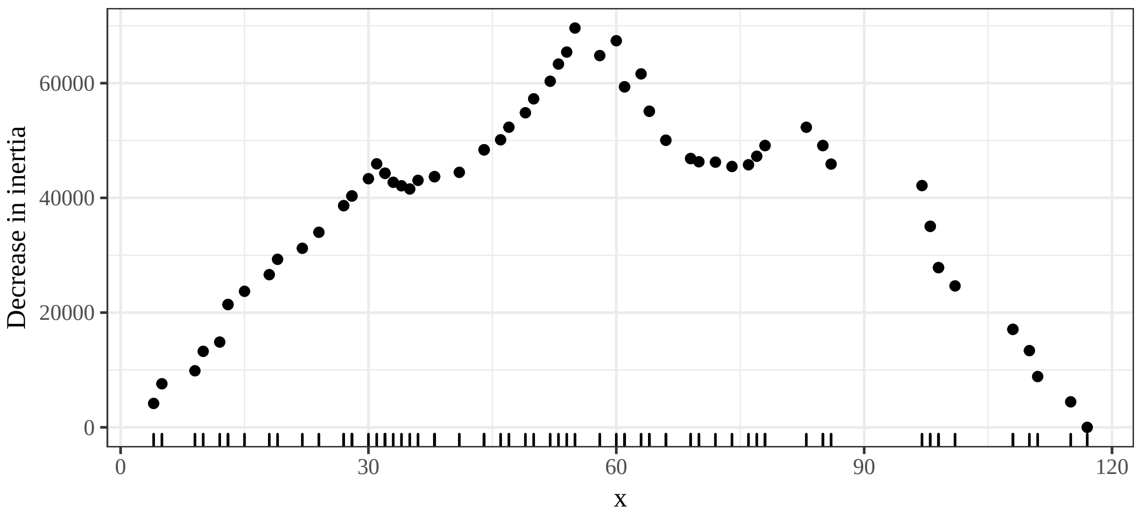


Figure 6.2: Decrease in inertia for each cutoff value in variable x in *ruspini*. There are three modes in the plot with two local maxima and one global maximum.

the decrease in inertia vs. cut points in the x variable of the example *ruspini* data. Therefore, the potential search algorithms should be able to handle this non-linear problem and be able to avoid locally optimal values. Some candidate algorithms that have been proposed are the monkey search (Chen et al., 2014), hill climbing hybrid with random walks, or simulated annealing (both are mentioned in Russell et al., 2010). However, using heuristic searches does not guarantee that the optimal split will be found, and the results are not stable because of the randomness in their algorithms. That is the reason why we still preferred the exhaustive search in this dissertation to assess the performance of the clustering techniques. However, for data with large n (such as ice extent data) or Q (like functional data), the speed of a heuristic algorithm can overshadow the accuracy.

BIBLIOGRAPHY

Adolfsson, A., Ackerman, M., and Brownstein, N. C. (2019). “To cluster, or not to cluster: An analysis of clusterability methods”. In: *Pattern Recognition* 88, pp. 13–26. ISSN: 0031-3203. DOI: 10.1016/J.PATCOG.2018.10.026.

Agostinelli, C. and Lund, U. (2017). *R package circular: Circular Statistics (version 0.4-93).*

Anderson, M. J. (2001). “A new method for non-parametric multivariate analysis of variance”. In: *Austral Ecology* 26.1, pp. 32–46. ISSN: 14429985. DOI: 10.1111/j.1442-9993.2001.01070.pp.x.

Arkhangel’skii, A. V. and Pontryagin, L. S. (1990). *General Topology I.* Ed. by A. V. Arkhangel’skii and L. S. Pontryagin. Vol. 17. Encyclopaedia of Mathematical Sciences. Berlin, Heidelberg: Springer Berlin Heidelberg. ISBN: 978-3-642-64767-3. DOI: 10.1007/978-3-642-61265-7.

Banfield, J. D. and Raftery, A. E. (1993). “Model-Based Gaussian and Non-Gaussian Clustering”. In: *Biometrics* 49.3, p. 803. ISSN: 0006341X. DOI: 10.2307/2532201.

Barbour, C., Kosa, P., Komori, M., Tanigawa, M., Masvekar, R., Wu, T., Johnson, K., Douvaras, P., Fossati, V., Herbst, R., Wang, Y., Tan, K., Greenwood, M., and Bielekova, B. (2017). “Molecular-based diagnosis of multiple sclerosis and its progressive stage”. In: *Annals of Neurology* 82.5, pp. 795–812. ISSN: 03645134. DOI: 10.1002/ana.25083.

Ben-David, S. (2015). “Computational Feasibility of Clustering under Clusterability Assumptions”. In: arXiv: 1501.00437.

Berry, K. J., Mielke, P. W., and Johnston, J. E. (2016). *Permutation Statistical Methods.* Cham: Springer International Publishing, p. 622. ISBN: 978-3-319-28768-3. DOI: 10.1007/978-3-319-28770-6.

Boik, R. J. (1987). “The Fisher-Pitman permutation test: A non-robust alternative to the normal theory F test when variances are heterogeneous”. In: *British Journal of Mathematical and Statistical Psychology* 40.1, pp. 26–42. ISSN: 20448317. DOI: 10.1111/j.2044-8317.1987.tb00865.x.

Bray, J. R. and Curtis, J. T. (1957). “An Ordination of the Upland Forest Communities of Southern Wisconsin”. In: *Ecological Monographs* 27.4, pp. 325–349. ISSN: 00129615. DOI: 10.2307/1942268.

Breiman, L., Friedman, J., Stone, C. J., and Olshen, R. (1984). *Classification and Regression Trees*. 1st ed. Chapman and Hall/CRC. ISBN: 0412048418.

Brunson, J. C. (2018). *ggalluvial: Alluvial Diagrams in 'ggplot2'*. R package version 0.9.1.

Caliński, T. and Harabasz, J (1974). “A dendrite method for cluster analysis”. en. In: *Communications in Statistics* 3.1, pp. 1–27.

Casella, G. and Berger, R. L. (2002). *Statistical inference*. 2nd. Thomson Learning, p. 660. ISBN: 9780534243128.

Celebi, M. E., Kingravi, H. A., and Vela, P. A. (2013). “A comparative study of efficient initialization methods for the k-means clustering algorithm”. In: *Expert Systems with Applications* 40.1, pp. 200–210. ISSN: 0957-4174. DOI: 10.1016/J.ESWA.2012.07.021.

Charrad, M., Ghazzali, N., Boiteau, V., and Niknafs, A. (2014). “NbClust : An R Package for Determining the”. In: *Journal of Statistical Software* 61.6.

Chavent, M. (1998). “A monothetic clustering method”. In: *Pattern Recognition Letters* 19.11, pp. 989–996. ISSN: 01678655. DOI: 10.1016/S0167-8655(98)00087-7.

Chavent, M., Lechevallier, Y., and Briant, O. (2007). “DIVCLUS-T: A monothetic divisive hierarchical clustering method”. In: *Computational Statistics & Data Analysis* 52.2, pp. 687–701. ISSN: 01679473. DOI: 10.1016/j.csda.2007.03.013.

Chen, X., Zhou, Y., and Luo, Q. (2014). “A hybrid monkey search algorithm for clustering analysis.” In: *TheScientificWorldJournal* 2014, p. 938239. ISSN: 1537-744X. DOI: 10.1155/2014/938239.

Clarkson, D. B., Fraley, C., Gu, C. C., and Ramsey, J. O. (2002). “Functional Cluster Analysis”. In: *S+ Functional Data Analysis*. Ed. by J. O. Ramsay and B. W. Silverman. Springer Series in Statistics. New York: Springer-Verlag, pp. 155–164. ISBN: 978-0-387-95414-1. DOI: 10.1007/0-387-28393-5_10.

Craven, P. and Wahba, G. (1978). “Smoothing noisy data with spline functions”. In: *Numerische Mathematik* 31.4, pp. 377–403. ISSN: 0029-599X. DOI: 10.1007/BF01404567.

Cuadras, C. and Arenas, C. (1990). “A distance based regression model for prediction with mixed data”. In: *Communications in Statistics - Theory and Methods* 19.6, pp. 2261–2279. ISSN: 0361-0926. DOI: 10.1080/03610929008830319.

De Boor, C. (1972). “On Calculating with Mplines”. In: *JOURNAL OF APPROXIMATION THEORY* 6, p. 62.

Dua, D. and Graff, C. (2019). *UCI Machine Learning Repository*. Irvine, CA.

Elvidge, A. D., Renfrew, I. A., Elvidge, A. D., and Renfrew, I. A. (2016). “The Causes of Foehn Warming in the Lee of Mountains”. In: *Bulletin of the American Meteorological Society* 97.3, pp. 455–466. ISSN: 0003-0007. DOI: 10.1175/BAMS-D-14-00194.1.

Esri (2018). *ArcGIS*. Redlands.

Everitt, B. and Hothorn, T. (2011). *An Introduction to Applied Multivariate Analysis with R*. 1st ed. Springer. ISBN: 1441996494.

Everitt, B. S., Landau, S., Leese, M., and Stahl, D. (2011). *Cluster Analysis*. 5th ed. Wiley, p. 346. ISBN: 0470749911.

Febrero-Bande, M. and Fuente, M. O. de la (2012). “Statistical Computing in Functional Data Analysis: The R Package fda.usc”. In: *Journal of Statistical Software* 51.4, pp. 1–28.

Fetterer, F., Knowles, F., Meier, W., Savoie, M., and Windnagel, A. K. (2018). *Sea Ice Index, Version 3*. DOI: 10.7265/N5K072F8.

Fisher, N. I. (1993). *Statistical Analysis of Circular Data*. Cambridge: Cambridge University Press. ISBN: 9780511564345. DOI: 10.1017/CBO9780511564345.

Fisher, R. A. (1936). “The Use of Multiple Measurements in Taxonomic Problems”. In: *Annals of Eugenics* 7.2, pp. 179–188. DOI: 10.1111/j.1469-1809.1936.tb02137.x.

Gower, J. C. (1971). “A General Coefficient of Similarity and Some of Its Properties”. In: *Biometrics* 27.4, p. 857. ISSN: 0006341X. DOI: 10.2307/2528823.

Greenwood, M. C. (2004). “Functional Data Analysis for Glaciated Valley Profile Analysis”. PhD thesis. University of Wyoming, Laramie.

Greenwood, M. C., Sojda, R. S., Sharp, J. L., Peck, R. G., and Rosenberry, D. O. (2011). “Multi-scale Clustering of Functional Data with Application to Hydraulic Gradients in Wetlands”. In: *Journal of Data Science* 9.3, pp. 399–426.

Gu, Z., Gu, L., Eils, R., Schlesner, M., and Brors, B. (2014). “circlize implements and enhances circular visualization in R”. In: *Bioinformatics* 30.19, pp. 2811–2812. ISSN: 1460-2059. DOI: 10.1093/bioinformatics/btu393.

Handl, J., Knowles, J., and Kell, D. B. (2005). “Computational cluster validation in post-genomic data analysis”. In: *Bioinformatics* 21.15, pp. 3201–3212. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/bti517.

Härdle, W. K. and Simar, L. (2015). *Applied Multivariate Statistical Analysis*. Berlin, Heidelberg: Springer Berlin Heidelberg. ISBN: 978-3-662-45170-0. DOI: 10.1007/978-3-662-45171-7.

Hardy, A. (1996). "On the number of clusters". In: *Computational Statistics & Data Analysis* 23.1, pp. 83–96. ISSN: 0167-9473. DOI: 10.1016/S0167-9473(96)00022-9.

Hartigan, J. A. and Hartigan, P. M. (1985). "The Dip Test of Unimodality". In: *The Annals of Statistics* 13.1, pp. 70–84. ISSN: 0090-5364. DOI: 10.1214/aos/1176346577.

Hartigan, J. A. and Wong, M. A. (1979). "Algorithm AS 136: A K-Means Clustering Algorithm". In: *Applied Statistics* 28.1, p. 100. ISSN: 00359254. DOI: 10.2307/2346830.

Hastie, T., Tibshirani, R., and Friedman, J. (2016). *The Elements of Statistical Learning*. 2nd ed. Springer. ISBN: 978-0387848570.

Heckman, N. and Zamar, R. (2000). "Comparing the shapes of regression functions". In: *Biometrika* 87.1, pp. 135–144. ISSN: 0006-3444. DOI: 10.1093/biomet/87.1.135.

Hitchcock, D. B. and Greenwood, M. C. (2015). "Clustering Functional Data". In: *Handbook of Cluster Analysis*. Ed. by C. Hennig, M. Meila, F. Murtagh, and R. Rocci. 1st ed. Vol. 1. Chapman and Hall/CRC. Chap. 13, pp. 265–288. ISBN: 9781466551893. DOI: 10.1201/b19706. arXiv: arXiv:1504.06917v1.

Hitchcock, D. B., Booth, J. G., and Casella, G. (2007). "The effect of pre-smoothing functional data on cluster analysis". In: *Journal of Statistical Computation and Simulation* 77.12, pp. 1043–1055. ISSN: 0094-9655. DOI: 10.1080/10629360600880684.

Hofmann, H. and Vendettuoli, M. (2013). “Common Angle Plots as Perception-True Visualizations of Categorical Associations”. In: *IEEE Transactions on Visualization and Computer Graphics* 19.12, pp. 2297–2305. ISSN: 1077-2626. DOI: 10.1109/TVCG.2013.140.

Hothorn, T., Hornik, K., and Zeileis, A. (2006). “Unbiased Recursive Partitioning: A Conditional Inference Framework”. en. In: *Journal of Computational and Graphical Statistics* 15.3, pp. 651–674. ISSN: 1061-8600. DOI: 10 . 1198 / 106186006X133933.

Hubert, L. and Arabic, P. (1985). “Comparing Partitions”. In: *Journal of Classification* 2, pp. 193–218.

Inselberg, A. and Dimsdale, B. (1987). “Parallel Coordinates for Visualizing Multi-Dimensional Geometry”. In: *Computer Graphics 1987*. Tokyo: Springer Japan, pp. 25–44. DOI: 10.1007/978-4-431-68057-4_3.

James, G., Witten, D., Hasite, T., and Tibshirani, R. (2013). *An Introduction to Statistical Learning: with Applications in R*. 1st. Springer, p. 426. ISBN: 1461471370.

James, G. M. and Sugar, C. A. (2003). “Clustering for Sparsely Sampled Functional Data”. In: *Journal of the American Statistical Association* 98.462, pp. 397–408. ISSN: 0162-1459. DOI: 10.1198/016214503000189.

Jammalamadaka, S. R. and SenGupta, A. (2001). *Topics in Circular Statistics*. Vol. 5. ISBN: 9789812779267. DOI: 10.1142/9789812779267.

Kaufman, L. and Rousseeuw, P. J. (1990). *Finding Groups in Data: An Introduction to Cluster Analysis*. 1st ed. Wiley-Interscience, p. 368. ISBN: 978-0471735786.

Lin, Y.-S., Heathcote, A., and Kvam, P. (2018). *CircularDDM: Circular Drift-Diffusion Model*. R package version 0.1.0.

Lloyd, S. (1982). “Least squares quantization in PCM”. In: *IEEE Transactions on Information Theory* 28.2, pp. 129–137. ISSN: 0018-9448. DOI: 10.1109/TIT.1982.1056489.

Loperfido, N. and Tarpey, T. (2018). “Some remarks on the <i>R</i> <sup>2</sup> for clustering”. In: *Statistical Analysis and Data Mining: The ASA Data Science Journal* 11.3, pp. 135–148. ISSN: 19321864. DOI: 10.1002/sam.11378.

Lund, U. J. (1999). “Least circular distance regression for directional data”. In: *Journal of Applied Statistics* 26.6, pp. 723–733. ISSN: 0266-4763. DOI: 10.1080/02664769922160.

— (2002). “Tree-Based Regression for a Circular Response”. In: *Communications in Statistics - Theory and Methods* 31.9, pp. 1549–1560. ISSN: 0361-0926. DOI: 10.1081/STA-120013011.

Maaten, L. van der and Hinton, G. (2008). “Visualizing High-Dimensional Data Using t-SNE”. In: *Journal of Machine Learning Research* 9.Nov, pp. 2579–2605. ISSN: 0885-6125.

MacNaughton-Smith, P., Williams, W. T., Dale, M. B., and Mockett, L. G. (1964). “Dissimilarity Analysis: a new Technique of Hierarchical Sub-division”. In: *Nature* 202.4936, pp. 1034–1035. ISSN: 0028-0836. DOI: 10.1038/2021034a0.

MacQueen, J. (1967). *Some methods for classification and analysis of multivariate observations*. Berkeley, Calif.

Maechler, M., Rousseeuw, P., Struyf, A., Hubert, M., and Hornik, K. (2018). *cluster: Cluster Analysis Basics and Extensions*. R package version 2.0.7-1 — For new features, see the ’Changelog’ file (in the package source).

Maslanik, J. A., Fowler, C., Stroeve, J., Drobot, S., Zwally, J., Yi, D., and Emery, W. (2007). “A younger, thinner Arctic ice cover: Increased potential for rapid, extensive sea-ice loss". In: *Geophysical Research Letters* 34.24, p. L24501. ISSN: 0094-8276. DOI: 10.1029/2007GL032043.

Michaud, A. B., Šabacká, M., and Priscu, J. C. (2012). "Cyanobacterial diversity across landscape units in a polar desert: Taylor Valley, Antarctica". In: *FEMS Microbiology Ecology* 82.2, pp. 268–278. ISSN: 01686496. DOI: 10.1111/j.1574-6941.2012.01297.x.

Milligan, G. W. and Cooper, M. C. (1985). "An examination of procedures for determining the number of clusters in a data set". In: *Psychometrika* 50.2, pp. 159–179. ISSN: 0033-3123. DOI: 10.1007/BF02294245.

Murtagh, F. and Legendre, P. (2014). "Ward's Hierarchical Agglomerative Clustering Method: Which Algorithms Implement Ward's Criterion?" In: *Journal of Classification* 31.3, pp. 274–295. ISSN: 0176-4268. DOI: 10.1007/s00357-014-9161-z.

Oksanen, J., Blanchet, F. G., Friendly, M., Kindt, R., Legendre, P., McGlinn, D., Minchin, P. R., O'Hara, R. B., Simpson, G. L., Solymos, P., Stevens, M. H. H., Szoecs, E., and Wagner, H. (2018). *vegan: Community Ecology Package*. R package version 2.5-2.

— (2019). *vegan: Community Ecology Package*. R package version 2.5-4.

Oliveira, M., Crujeiras, R. M., and Rodríguez-Casal, A. (2014). "NPCirc: An R Package for Nonparametric Circular Methods". In: *Journal of Statistical Software* 61.9, pp. 1–26. DOI: 10.18637/jss.v061.i09.

Pavoine, S., Vallet, J., Dufour, A.-B., Gachet, S., and Daniel, H. (2009). "On the challenge of treating various types of variables: application for improving the measurement of functional diversity". In: *Oikos* 118.3, pp. 391–402. ISSN: 00301299. DOI: 10.1111/j.1600-0706.2008.16668.x.

Pewsey, A., Neuhäuser, M., and Ruxton, G. D. (2013). *Circular Statistics in R.* Oxford University Press, p. 183. ISBN: 9780199671137.

Piccarreta, R. and Billari, F. C. (2007). "Clustering work and family trajectories by using a divisive algorithm". In: *Journal of the Royal Statistical Society: Series A (Statistics in Society)* 170.4, pp. 1061–1078. ISSN: 0964-1998. DOI: `10.1111/j.1467-985X.2007.00495.x`.

Pison, G., Struyf, A., and Rousseeuw, P. J. (1999). "Displaying a clustering with CLUSPLOT". In: *Computational Statistics & Data Analysis* 30.4, pp. 381–392. ISSN: 01679473. DOI: `10.1016/S0167-9473(98)00102-9`.

R Core Team (2019). *R: A Language and Environment for Statistical Computing.* R Foundation for Statistical Computing. Vienna, Austria.

Ramsay, J. O. and Silverman, B. W. (2005). *Functional data analysis.* Springer, p. 426. ISBN: 9780387400808.

Ramsay, J. O., Wickham, H., Graves, S., and Hooker, G. (2018). *fda: Functional Data Analysis.* R package version 2.4.8.

Rand, W. M. (1971). "Objective Criteria for the Evaluation of Clustering Methods". In: *Journal of the American Statistical Association* 66, pp. 846–850. ISSN: 0162-1459. DOI: `10.1080/01621459.1971.10482356`.

Rousseeuw, P. J. (1987). "Silhouettes: A graphical aid to the interpretation and validation of cluster analysis". In: *Journal of Computational and Applied Mathematics* 20, pp. 53–65. ISSN: 03770427. DOI: `10.1016/0377-0427(87)90125-7`.

Ruspini, E. H. (1970). "Numerical methods for fuzzy clustering". In: *Information Sciences* 2.3, pp. 319–350. ISSN: 00200255. DOI: `10.1016/S0020-0255(70)80056-1`.

Russell, S. J. S. J., Norvig, P., and Davis, E. (2010). *Artificial intelligence : a modern approach*. 3rd ed. Pearson, p. 1132. ISBN: 9780136042594.

Šabacká, M., Priscu, J. C., Basagic, H. J., Fountain, A. G., Wall, D. H., Virginia, R. A., and Greenwood, M. C. (2012). “Aeolian flux of biotic and abiotic material in Taylor Valley, Antarctica”. In: *Geomorphology* 155-156, pp. 102–111. ISSN: 0169555X. DOI: `10.1016/j.geomorph.2011.12.009`.

Serreze, M. C., Maslanik, J. A., Scambos, T. A., Fetterer, F., Stroeve, J., Knowles, K., Fowler, C., Drobot, S., Barry, R. G., and Haran, T. M. (2003). “A record minimum arctic sea ice extent and area in 2002”. In: *Geophysical Research Letters* 30.3, p. 1110. ISSN: 0094-8276. DOI: `10.1029/2002GL016406`.

Serreze, M. C., Holland, M. M., and Stroeve, J. (2007). “Perspectives on the Arctic’s shrinking sea-ice cover.” In: *Science (New York, N.Y.)* 315.5818, pp. 1533–6. ISSN: 1095-9203. DOI: `10.1126/science.1139426`.

Simpson, G. L. (2016). *permute: Functions for Generating Restricted Permutations of Data*. R package version 0.9-4.

Sneath, P. H. A. and Sokal, R. R. (1973). *Numerical taxonomy: the principles and practice of numerical classification*. W.H. Freeman, p. 573. ISBN: 0716706970.

Suarez, A. J. and Ghosal, S. (2016). “Bayesian Clustering of Functional Data Using Local Features”. In: *Bayesian Analysis* 11.1, pp. 71–98. ISSN: 1936-0975. DOI: `10.1214/14-BA925`.

Tarpey, T. (2007). “Linear Transformations and the k -Means Clustering Algorithm”. In: *The American Statistician* 61.1, pp. 34–40. ISSN: 0003-1305. DOI: `10.1198/000313007X171016`.

Tarpey, T. and Kinateder, K. K. J. (2003). “Clustering Functional Data”. In: *Journal of Classification* 20.1, pp. 93–114. ISSN: 0176-4268. DOI: `10.1007/s00357-003-0007-3`.

Therneau, T. and Atkinson, B. (2018). *rpart: Recursive Partitioning and Regression Trees*. R package version 4.1-13.

Tibshirani, R., Walther, G., and Hastie, T. (2001). “Estimating the number of clusters in a data set via the gap statistic”. In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 63.2, pp. 411–423. ISSN: 1369-7412. DOI: 10.1111/1467-9868.00293.

Tuddenham, R. D. and Snyder, M. M. (1954). “Physical growth of California boys and girls from birth to eighteen years.” In: *Publications in child development. University of California, Berkeley* 1.2, pp. 183–364.

Vsevolozhskaya, O., Greenwood, M., and Holodov, D. (2014). “Pairwise comparison of treatment levels in functional analysis of variance with application to erythrocyte hemolysis”. In: *Annals of Applied Statistics* 8.2, pp. 905–925. ISSN: 19417330. DOI: 10.1214/14-AOAS723. arXiv: arXiv:1407.8388v1.

Wang, S., Jank, W., and Shmueli, G. (2008). “Explaining and Forecasting Online Auction Prices and Their Dynamics Using Functional Data Analysis”. In: ISSN: 1537-2707. DOI: 10.1198/073500106000000477.

Ward, J. H. (1963). “Hierarchical Grouping to Optimize an Objective Function”. In: *Journal of the American Statistical Association* 58.301, pp. 236–244. ISSN: 0162-1459. DOI: 10.1080/01621459.1963.10500845.

Wetlaufer, K., Hendrikx, J., and Marshall, L. (2016). “Spatial Heterogeneity of Snow Density and Its Influence on Snow Water Equivalence Estimates in a Large Mountainous Basin”. In: *Hydrology* 3.1, p. 3. ISSN: 2306-5338. DOI: 10.3390/hydrology3010003.

Wickham, H. (2016). *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York. ISBN: 978-3-319-24277-4.

Will, G. (2016). “Visualizing and Clustering Data that Includes Circular Variables”. Writing Project. Montana State University.

Witten, D. M. and Tibshirani, R. (2010). “A framework for feature selection”. In: *American Statistician* 105.490, pp. 713–726. ISSN: 0162-1459. DOI: `10.1198/jasa.2010.tm09415.A`.

Wood, S. (2006). *Generalized Additive Models: An Introduction with R*. 1st ed. Chapman and Hall/CRC. ISBN: 1584884746.

Zajacova, A., Huzurbazar, S., Greenwood, M., and Nguyen, H. (2015). “Long-Term BMI Trajectories and Health in Older Adults”. In: *Journal of Aging and Health* 27.8, pp. 1443–1461. ISSN: 0898-2643. DOI: `10.1177/0898264315584329`.