Exploration of Cross-Validation For Selecting the Number of Clusters

TAN VINH TRAN

Department of Mathematical Sciences
Montana State University

May 3, 2014

A writing project submitted in partial fulfillment
of the requirements for the degree

Master of Science in Statistics

APPROVAL

of a writing project submitted by

TAN VINH TRAN

This writing project has been read by the writing project advisor and has been found to be satisfactory regarding content, English usage, format, citations, bibliographic style, and consistency, and is ready for submission to the Statistics Faculty.

Date	Mark Greenwood Writing Project Advisor
Date	Megan D. Higgs Writing Projects Coordinator

Contents

List of Figures

List of Tables

1 Motivation

Grouping similar objects into meaningful categories is fundamental in many sciences, such as biology, astronomy, computer science, etc. The main purpose of classification is usually finding the *similarities* within groups and *differences* between groups to understand their characteristics. Another purpose is based on those characteristics, one can *predict* which group a new object belongs to. Numerical techniques for objectively and stably classifying objects are called *cluster analysis* (Everitt *et al.*, 2011). Since the correct number of groups and group membership is unknown before classification, cluster analysis is categorized as *unsupervised leaning.*

A case study is a good place to start to understand the methods. The classification of wheat kernels (Bache & Lichman, 2013) based on their characteristics are investigated as an example of a cluster analysis problem. The data set comprises 210 X-ray measurements for randomly selected kernels from three different varieties of wheat: Kama, Rosa and Canadian. The information from the wheat kernels includes: area, perimeter, compactness, length of kernel, width of kernel, asymmetry coefficient and length of kernel groove. Since these are measurements on the different aspects of the same kernels, high collinearity among many of the variables are expected. The biplot in Fig. 1 confirms this. Two groups of variables can be seen in the plot. The kernel's dimension-related measurements, namely length, width, perimeter and area of kernels and length of groove are positively correlated with each other. Compactness and asymmetry vectors create an angle close to 180^o, suggesting they are strongly negatively correlated with each other. These two groups of variables have low collinearity with each other. An excerpt from the data set can been seen in Table 1.

In order to display the data set in two dimensions, Principal Component Analysis (PCA) is applied to the variables. Basically, *principal components* (PC) are linear combinations of the original variables such that PCs are uncorrelated and are ordered so that the first few retain most of the variation from all the original variables (Fig.

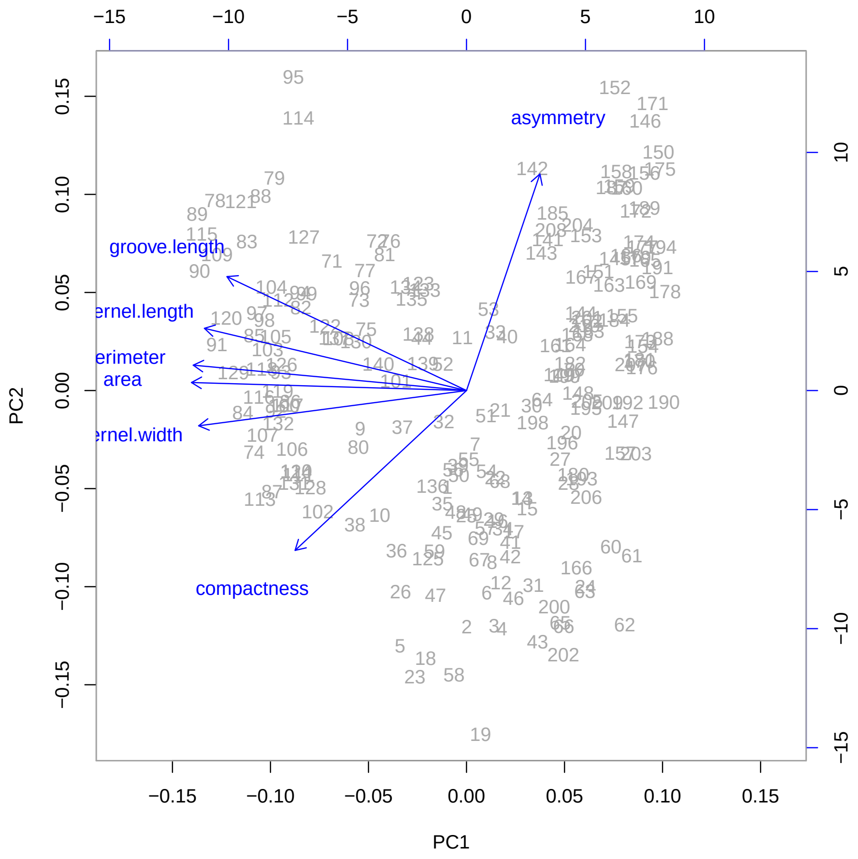


Figure 1: Biplot of Wheat kernels data set. The vectors represents variables in the data set, and gray numbers are observations. The cosine of the angle is the correlation.

2). The first two PCs of PCA on the data set explain almost 98% of the variability of the data, so the relative Euclidean distance between data points can also be validly observed in the biplot in Fig. 1.

The research questions that can be asked here are: (1) How many groups should the seeds be classified into? and (2) with that number of groups, which observations belong to the groups? and (3) what are the characteristics of each group?

This project will focus on the first question. The answer to the first question may also rely on the clustering algorithm which can also impact the results for the second and third questions.

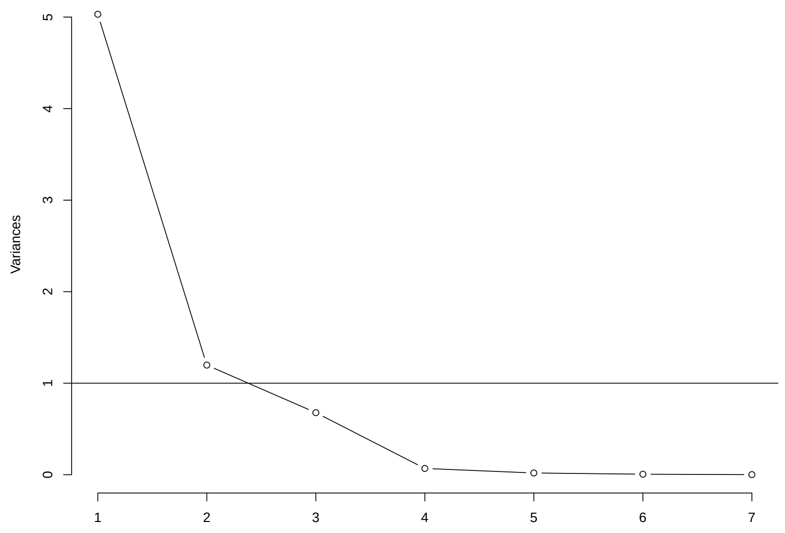


Figure 2: Screeplot showing the variance of original variables in kernel data set against the number of the principal components. The horizontal line at variance 1 is a reference line indicating that the first two components may be all that we needed here.

Table 1: An excerpt from the wheat kernels data set ($n = 210$) (Bache & Lichman, 2013)

Kernel #	area	perimeter	compactness	kernel.length	kernel.width	asymmetry	groove.length	variety
1	15.26	14.84	0.87	5.76	3.31	2.22	5.22	1
2	14.88	14.57	0.88	5.55	3.33	1.02	4.96	1
3	14.29	14.09	0.90	5.29	3.34	2.70	4.83	1
4	13.84	13.94	0.90	5.32	3.38	2.26	4.80	1
5	16.14	14.99	0.90	5.66	3.56	1.35	5.17	1
6	14.38	14.21	0.90	5.39	3.31	2.46	4.96	1
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

2 Methods

2.1 Cluster Analysis

As introduced in Section 1, Cluster Analysis (or Clustering) encompasses a set of numerical techniques whose the aim is to provide objective and stable classifications. Said in other words, the same numerical technique applied to a similar data set should produce similar classifications. Basically, clusters are defined in terms of internal cohesion (*homogeneity*) and external isolation (*separation*) (Everitt *et al.*, 2011). However, different concepts of homogeneity and separation create numerous and diverse clustering methods and potential results.

The important characteristic of cluster analysis problems is that the groups are unknown *a priori*. This is the reason why cluster analysis is categorized as *unsupervised learning* and often performed as part of *exploratory data analysis*. Cluster analysis is essentially about *discovering* groups in the data. Therefore, the analysis may be the only option when there are suspected groups in the data but no single variable is available to expose those groups. Unfortunately, it will also impose grouping where none may really exist (Greenwood, 2014).

A vast variety of clustering methods have been developed over recent decades and the results of cluster analysis are very dependent on the chosen methods. In the scope of this project, we decided to explore one specific set of popular clustering approaches for our purpose. The available options will be briefly discussed before introducing our choice of technique in the following sections.

2.1.1 Hierarchical clustering

The two best-known clustering approaches probably are *K-means clustering* and *hierarchical clustering*. K-means algorithm uses a predefined number of clusters to optionally partition the observations into those clusters. In the hierarchical clustering algorithm, on another hand, clusters at all sizes (1 to n) are created and the user then decides on how many clusters to report. While both methods have advantages and disadvantages, this project focuses on hierarchical clustering.

Hierarchical classification consists of a series of partitions, which may run from a single cluster containing all observations, all the way to n clusters each containing a single individual. This technique can be displayed in an attractive, tree-based representation of the observations, called a *dendrogram*. An example dendrogram from the wheat kernel problem can be seen in Fig. 3. At the bottom of the tree, each *leaf* represents one of 210 observations in the data set. As we move up the tree, the leaves which are similar to each other are grouped into *branches*. The branches then define combinations of observations until the top, where there is only one *root*. The height of the dendrogram indicates the *similarity* between observations or clusters of observations. Two observations meeting at a lower branch will be more similar to each other than another pair of observations meeting at a higher branch.

Hierarchical clustering techniques may be subdivided into *agglomerative* methods, which start from n individuals and successively fuse them into groups, and *divisive* methods, which separate the n individuals successively into smaller groups. From the perspective of the dendrogram, agglomerative methods start from the bottom and proceed to the top while divisive methods start from the root and go all the way to bottom leaves. That is why these methods are also described as *bottom up* and *top down*, respectively. In this project, we explore agglomerative methods. The choice is merely based on preferences, not on any specific advantages or disadvantages of the methods.

2.1.2 Clustering Algorithm

The algorithm of agglomerative clustering is very simple and detailed in Algorithm 1. A prerequisite for the algorithm is defining some sort of *dissimilarity* measure between

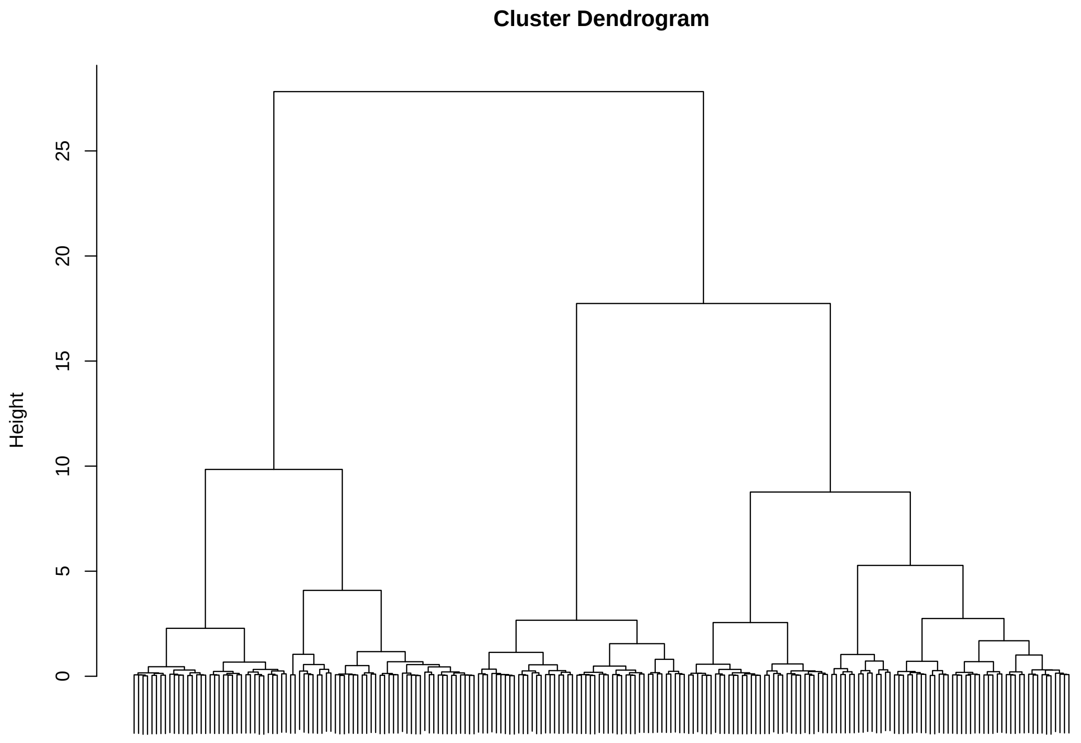


Figure 3: Dendrogram for Wheat kernel data set using agglomerative hierarchical methods with Ward's method algorithm.

each pair of observations. One of the most often used measures, Euclidean distance, is employed in this project. The Euclidean distance d_{ij} between two observations i and j in a data set with p variables is calculated by

$$d_{ij} = \sqrt{\sum_{k=1}^{p} (y_{ik} - y_{jk})^2}. \tag{1}$$

Apart from being a distance metric, which means this measure has a nice set of properties, Euclidean distances correspond to results from least squares models and are also directly related to the methods for choosing the number of clusters, discussed in later sections.

```
  Data: $n$ observations and a measure (such as Euclidean distance) of all
        $\binom{n}{2} = n(n-1)/2$ pairwise dissimilarities. Treat each observation as its
        own cluster.
  Result: Dendrogram
1 for $i = n$ to 2 do
2     examine all pairwise inter-cluster dissimilarities among $i$ clusters and
      identify the least dissimilar pair of clusters;
3     fuse these two clusters;
4     compute the new pairwise inter-cluster dissimilarities among the $i-1$
      remaining clusters;
5 end for
```

Algorithm 1: Hierarchical Clustering

Defining the dissimilarities between pairs of observations is only the first half of the story. The other question raised from the algorithm is how do we define the dissimilarity between two clusters if one or both of the clusters contain multiple observations? The answer can be achieved by developing the notion of *linkage*, which defines the dissimilarity between two groups of observations. There are some common types of linkage – *complete, average, single, centroid*, etc. Ward (1963) introduced a method in which the fusion of two clusters is based on the size of an error sum of squares criterion. The objective at each stage (for k clusters) is to minimize the increase in the total within-cluster error sum of squares, E, given by

$$E = \sum_{k=1}^{K} E_k,$$

where

$$E_k = \sum_{j=1}^{p} \sum_{i=1}^{n_k} (y_{ij,k} - \bar{y}_{\bullet j,k})^2, \tag{2}$$

in which $\bar{y}_{\bullet j,k} = (1/n_k) \sum_{i=1}^{n_k} y_{ij,k}$ (the mean of the kth cluster for the jth variable), $y_{ij,k}$ being the value on the jth variable ($j = 1, \dots, p$) for the ith object ($i = 1, \dots, n_k$) in the kth cluster ($k = 1, \dots, K$). The objective of Ward's method ties directly to an impurity measure in cross-validation, discussed below.

2.2 Classification and Regression Trees

Hierarchical clustering produces and makes use of a dendrogram, which is basically a tree presentation of results. Moreover, the problem of deciding how big the tree should be in classification and regression trees (C&RT) is similar to the problem of deciding how many clusters should be retained in cluster analysis (or choosing where to cut the dendrogram in hierarchical clustering). Therefore, it is worthwhile to briefly mention C&RT in this project to have an overview of the similarities and differences between two methods. From that, the reasons for applying the pruning method in C&RT to clustering will be discussed.

Essentially, C&RT (James *et al.*, 2013) is a *supervised learning* approach, in which they explain variation of one or many response variable(s) by one or more explanatory variables. If there is one response variable and it is categorical, we have a *classification tree*. If the response is numerical, it is called *regression tree*. An expansion of classification and regression trees to a set of multivariate response is called a *multivariate regression tree* (De'ath, 2002). All of these are constructed by recursively splitting the data using a simple rule based on an explanatory variable, one at a time. The objective of C&RT is dividing the response into mutually exclusive, homogeneous groups but also to keep the tree reasonably small. An example of a classification tree can be seen in Fig. 4, fit using `rpart` (Therneau *et al.*, 2014) in R.

Hierarchical clustering and C&RT share the same goal in dividing data into similar groups of similarity and both of them use trees as the display for the methods. But there are basic differences that separate the two approaches. Hierarchical clustering uses a dissimilarity measure between observations as the criterion to fuse the data, which means every variable in the data set is used for splitting, and the objective function is also based on this measure. Meanwhile, C&RT uses a set of explanatory variables to generate splitting rules, with the objective function based on the response variable(s). Multivariate regression trees are more similar to clustering, but with

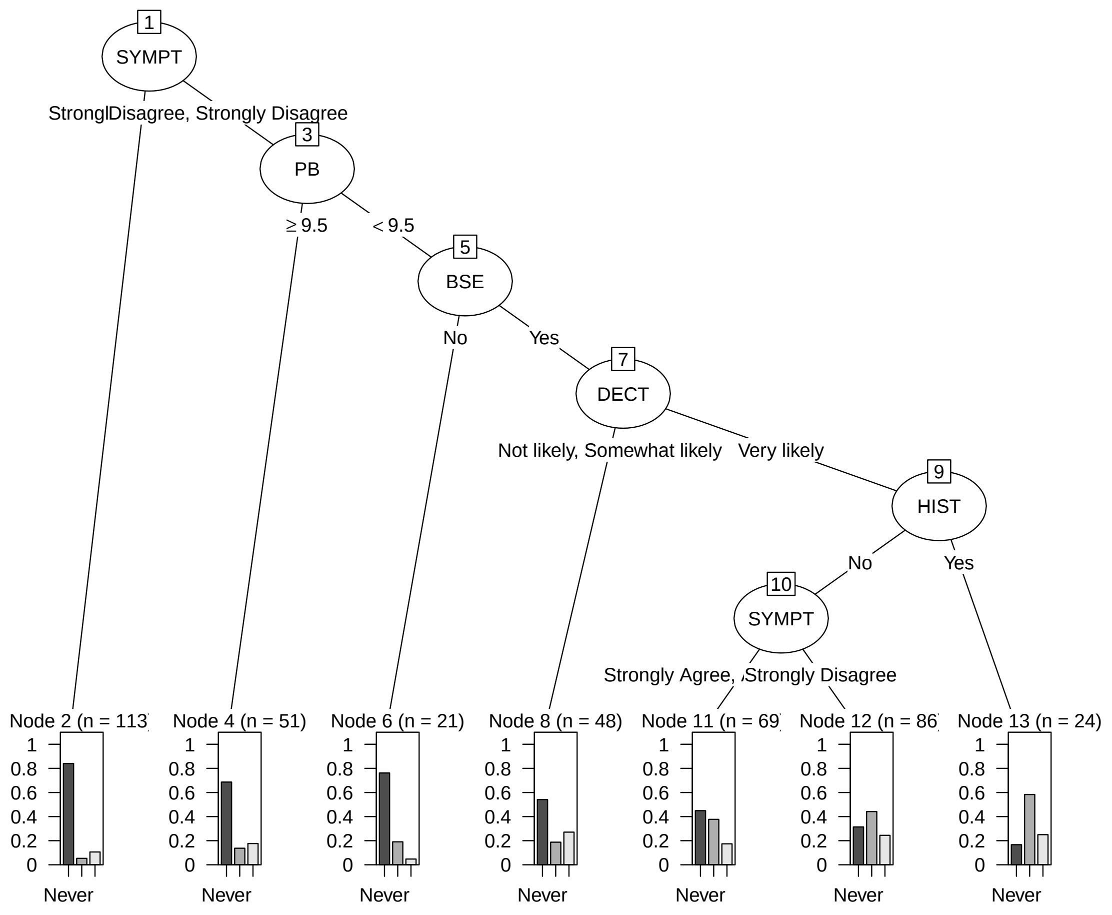


Figure 4: An example of classification tree. Response variable is a multinomial with three levels Never, Within a year and Over a year. The splitting rules are based on explanatory variables and shown clearly on each branch. In this tree, there are 7 groups of observations with similar results from response variable.

explanatory variables used to create groups. C&RT is considered a modeling method with two objectives: description and prediction, while clustering can only be used for exploratory purposes.

In C&RT, like any modeling, trees should not only explain variability in responses, but also need to have a reasonable size to be able to efficiently predict new, unobserved responses. One of many ways to decide the length of tree is based on finding continued improvement of the total impurity of the tree. This method usually creates a too-complicated tree and its prediction performance will suffer (De'ath & Fabricius, 2000). Hence, it is usually used as the first constraint to build a very complicated tree and then some sort of validation process is used to prune the tree to a smaller size. Because prediction is an important ability of C&RT, optimizing predictive error is a reasonable criterion in pruning trees. It can be achieved by using a validation data set or using cross-validation. Cross-validation is suitable when we do not have enough data to withhold a validation data set, but still provides a reasonable approximation of the test error (James *et al.*, 2013).

Because of the aforementioned similarity between pruning in C&RT and choosing the number of clusters in hierarchical cluster analysis, cross-validation seems to be a potential criterion to adapt to our cluster analysis problem. The details of cross-validation and how it can be applied to cluster analysis are mentioned in the next section.

2.3 Cross Validation

When we want to verify the predictive ability of a model, one way of doing this is to randomly split the data set into training and validation sets. The validation set will then be used to calculate the prediction error of the model fitted from the training set. In the absence of a very large data set to create a separate validation set, a number of techniques can be used as alternatives to estimate this quantity only using the available training data. Cross validation is among those techniques.

Instead of splitting the full data set into two separate sets, training and validating, cross-validation randomly divides the data set into subsets, holds a subset as the validating data and uses the rest of the data as the training set. The statistical model will be fitted on the training observations and a prediction of withheld observations is made. Since the withheld observations are not used in the fitting process, the mean squared error (MSE) of the observed and predicted values for the withheld set provides an approximately unbiased estimate for the test error. This process is repeated for each subset in the data set and the average of these test errors is the cross-validation estimate of the mean squared error of predicting a new observation.

The number of randomly divided equal-sized subsets from the data set can be different based on the needs of the researcher. If there are k subsets ($k < n$, where n is the number of observations in the data set), then the technique is called *k-fold cross validation.* When n subsets are used, or in other words, one observation will be withheld each time, it becomes *n-fold cross validation* or *Leave-one-out cross-validation* (LOOCV). The LOOCV estimate for the test MSE is the average of n test error estimates:

$$V_o = \frac{\sum (y_i - \hat{y}_i^{[-i]})^2}{n}, \tag{3}$$

where $\hat{y}_i^{[-i]}$ denotes the predicted value of y_i obtained from the model fitted to all data except y_i. The above formula can be extended to the multivariate case, where the responses are available on more than one variable:

$$V_m = \frac{\sum_{j=1}^{p} \sum_{i=1}^{n} (y_{ij} - \hat{y}_{ij}^{[-i]})^2}{n}, \tag{4}$$

where $\hat{y}_{ij}^{[-i]}$ is the predicted value of the jth variable of the ith observation from the model fitted to all data except the ith observation.

LOOCV has some major advantages over the separate validation set approach. First of all, LOOCV creates a "validating set" with virtually same size as the training data set, while in the validation set approach, the size of the testing data set is only, say, half of the original data set. Hence, LOOCV tends not to overestimate the test error rate (James *et al.*, 2013). Secondly, there is no randomness in LOOCV, which makes LOOCV perform the same every time we run it.

A typical LOOCV involves predicting withheld observations, which is not an intuitive process in cluster analysis. There is no splitting rule in typical clustering to predict which cluster a new observation should belong to. If we impose an assumption that an observation still lies in the same cluster it used to belong to before being withheld, this prediction difficulty can be overcome. With this assumption, there is no need to follow the process of withholding, predicting and calculating MSE for observations because we can use another formula, which is proved in Appendix A to be equal to Formula 4,

$$V_o = \frac{1}{n}\sum_{j=1}^{p}\sum_{i=1}^{n}\frac{(y_{ij}-\hat{y}_{ij})^2}{(1-A_{ii})^2}. \tag{5}$$

This formula, as described by Wood (2006) for a univariate response, consists of $\hat{y}_{ij}$, the predicted value of the jth variable of the ith observation from the model fitted to the *full* data set (no withholding). It also introduces a new denominator including A_{ii}, the diagonal item of the hat matrix $\mathbf{A}$ obtained from the cluster structure of the data set. Specifically, when we know the cluster that observations belong to, we can set up the model matrix as in a linear model, using the cluster information as a k-level categorical variable to generate X,

$$\mathbf{X} = \begin{pmatrix} 1 & 0 & \dots & 0 \\ 1 & 0 & & 0 \\ 1 & 1 & & 0 \\ 1 & 1 & & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 0 & & 1 \\ 1 & 0 & \dots & 1 \end{pmatrix},$$

and then calculate the hat matrix using (Robison-Cox, 2012)

$$\mathbf{A} = \mathbf{X}(\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top.$$

It should be noted that Formula 5 involves calculations over every single variable of observations. Because the denominators are different for each observation, the value needs to be calculated for each observation. For clustering with a large data set on many dimensions, this process involves a noticeable computational burden. Another type of cross validation introduced by Craven & Wahba (1978) – *Generalized cross-validation* (GCV) – which is popular in Generalized Additive Models, can be used to overcome this computational difficulty. GCV has the formula

$$V_g = \frac{\frac{1}{n}\sum_{j=1}^{p}\sum_{i=1}^{n}(y_{ij} - \hat{y}_{ij})^2}{[1 - \sum A_{ii}/n]^2}. \tag{6}$$

The improvement of this formula compared to Formula 5 (from now on will be called *Ordinary cross-validation* – OCV for differentiation purposes) is that the denominator is based on the average leverage calculated from the same hat matrix **A**. This leverage is constant over a given cluster structure (note that if we have k clusters, $\sum A_{ii} = k$, because $\sum A_{ii} = \text{tr}(A) = \text{rank}(X)$), thus leaving room for a workaround formula to calculate the MSE on the numerator as a whole without running through the inner iteration on the observations.

Cluster analysis often only requires the dissimilarity measure between pairs of observations, and the analyst may not have access to the raw data set. Therefore, a different formula to calculate the residual sum of square values on the numerator of Formula 6 without using the observed values, y_{ij}, and predicted values, $\hat{y}_{ij}$, is potentially useful. De'ath (2002), in a paper on multivariate regression trees, introduced a set of formulas showing the equivalence between observed data and distance matrices in calculating the residual sum of squares and prediction error. The proofs in Appendix A and B explore these results, including a correction to the published result. Specifically, we found that his result on Impurity (page 1116) listed as $\sum_{i<j,j} d_{ij}^2$ should be $\sum_{i<j,j} d_{ij}^2/n$ for the result to hold.

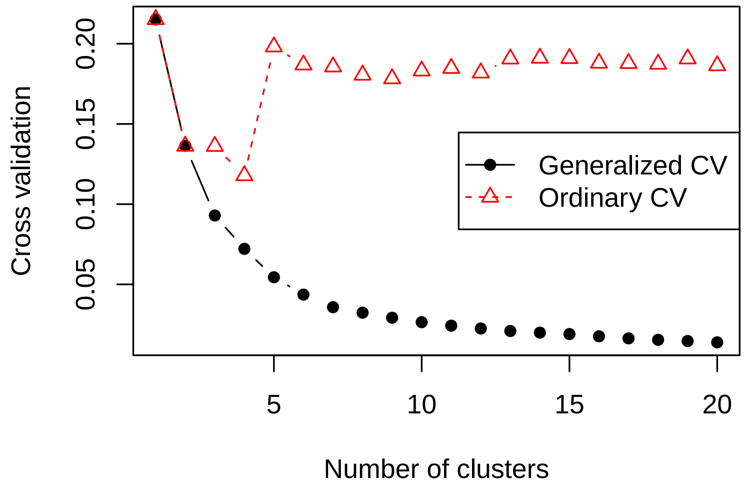


Figure 5: OCV and GCV for variety of clusters of Wheat kernels data set.

The relationship between total residual sum of square (RSS) and the dissimilarity matrix based on Euclidean distance is

$$\sum_{j=1}^{p}\sum_{i=1}^{n}(y_{ij}-\hat{y}_{ij})^2 = \frac{1}{n}\sum_{i<j,j} d_{ij}^2. \tag{7}$$

This formula is also considered as a "workaround" to calculate total RSS as a whole without going through $n \times p$ iterations. Hence it can improve the performance of the calculation enormously where there are many variables or clusters and the distance matrix is already available.

2.4 Selection Rules

OCV and GCV are only reference values for the potential number of clusters, as seen in the example of Wheat kernel data set in Fig. 5. Selection rules based on these criteria are needed to decide which number of clusters should be chosen. In this project, we explored three selection rules, one rule is based on OCV, and two rules are based on GCV.

2.4.1 Minimum Ordinary Cross Validation

The first rule is based on OCV, as indicated by the name, Minimum OCV. Picking the number of clusters using the minimum OCV makes sense because it is picking the solution with the smallest estimated MSEP. Unfortunately, the OCV estimator is more variable than the GCV and can sometimes miss the "correct" solution. Fig. 5 confirms that OCV values are quite wiggly across different cluster sizes but it does seem to have a clear minimum at 4 clusters. This suggests the usage of minimum OCV is a "good" cluster solution.

2.4.2 Generalized Cross Validation Based Rules

From Formula 6, we can infer a characteristic of GCV when the number of clusters increase. As the number of clusters k increases, the numerator (total residual sum of squares) decreases when using Ward's method, while the denominator increases, making GCV decrease gradually. Figure 5 is a typical example for the gradual decrease of GCV. When this happens, the minimum value of GCV is always the highest number of clusters considered, which makes choosing the minimum GCV a poor method. If the denominator increased more rapidly, then this measure might show an earlier minimum. Other approaches need to be considered in order to get better performance with GCV.

In statistical techniques involving choosing the number of variables retained such as choosing the number of principal components in PCA, or choosing the number of factors in EFA, a plot like Fig. 5 (which is called a scree plot) is made and the "elbow" is used to select the number of components (Everitt & Hothorn, 2011). The "elbow" in a scree plot is in fact the point where the value changes the most. This idea is adapted to the GCV measure to find the "elbow" in the cross validation plot.

The first rule to find the elbow is **relative slope**, in which the elbow is decided as the point where the change in slope is “steep” enough to support that the additional components do not provide “enough” extra improvement to include them. There is no exact rule to decide how steep is “enough”. The suggested rule used in this project is that in order to be considered steep, we need to make sure that the changing of GCV is a “large” percentage of the $k = 1$ result. The threshold is set to be 10% of the GCV of the one-cluster case as the default value. Although “ten percent” is an ad-hoc choice, it is proved to produce good results in our simulation study (Section 3). The rule can be formulated as

$$\text{if } \frac{GCV(i+1) - GCV(i)}{GCV(1)} < 0.1, \text{ then stop.}$$

Another approach to pick the “elbow” in the cross validation is using the definition of *large changes in slope*, which is basically the second derivative of the function. This approach is called *acceleration factor*. This method was one of a set of proposed methods used by Raîche *et al.* (2013) for selecting the number of factors in exploratory factor analysis. The formula to be used in this project is

$$\max_i \{ \quad GCV(i+1) - 2 \cdot GCV(i) + GCV(i-1) \quad \}. \tag{8}$$

This formula is a modified version from the original paper. This corrects an apparent mistake in their original paper on the approximation of second derivative formula. We also decided to change the selection rule to $\max_i$, instead of $\max_i -1$ as in the paper to pull the chosen point to the exact maximum point instead of the point before it. Changing the rule like this helps to improve the results of the chosen number of clusters in Section 3, but it also has a drawback that this rule cannot ever select one cluster. This is because in order for the formula to be meaningful, i has to be greater than 1.

3 Simulation Study

In order to see how cross validation with different selection rules performs, we set up a simulation study including a variety of schemes ranging from different number of dimensions to different separation of clusters.

3.1 Simulation Data

The data for the simulation study were created following six different settings, inspired by Tibshirani *et al.* (2001) and Walesiak & Dudek (2014):

1. *Null model in 10 dimensions* – observations are uniformly distributed within the range (-1, 1) in 10 dimensions;

    - A special case with virtually one big cluster.

2. *Three clusters model* – three clusters have bi-variate normal distributions with mean $(0, 0)$, $(0, 5)$ and $(5, -3)$ and variance $\mathbf{I}$. Cluster sizes are (25, 25, 50) respectively;

    - A easy case with three well-separated clusters.

3. *Random 4-cluster model in 3 dimensions* – four clusters that have standard normal distributions. Mean values are sampled from a $N(0, 5\mathbf{I})$ distribution and cluster sizes are randomly chosen from either 25 or 50 observations. Every data set in which the distance between any pairs of cluster is less than 2 was discarded from the simulations;

    - A difficult case, because the clusters are not well-separated (the smallest Euclidean distance between points in any two clusters are only 2 units).

4. *Random 4-cluster model in 10 dimensions* – four clusters that have standard normal distributions. Mean values are sampled from a $N(0, 1.9\mathbf{I})$ distribution and cluster sizes are randomly chosen from either 25 or 50 observations. Similar

to the previous case, every data set in which the distance between any pairs of clusters are less than 2 was discarded from the testing data;

- Even more difficult case. Although the minimum distance is the same as the previous case, increasing the dimensions makes the points in different clusters closer, because the distance is the sum over all dimensions.

5. *Two elongated clusters in three dimensions* – vector of a hundred values are sampled from a uniform distribution from -0.5 to 0.5, called t. Three variables are created so that $x_1 = x_2 = x_3 = t$, then added a random noise value from $N(0, 0.1\mathbf{I})$. The second cluster follows the same setting but all observations are shifted 10 units.

   - This can be considered the easiest case among the simulations. The clusters are very far away from each other, and distributed along the diagonal of the 3D cube.

6. *Four well separated clusters in 3 dimensions* – the observations are independently drawn from univariate normal distribution with means $(-4, 5, -4)$, $(5, 14, 5)$, $(14, 5, 14)$, $(5, -4, 5)$, and identity covariance matrix Σ.

   - This setting is an expansion of setting 2. This scenario is created to assess whether the rules tend to choose a fixed number of clusters instead of getting close to the designed number of clusters.

In each setting, 50 duplicate data sets are created. Examples of data sets from each setting are shown in Fig. 6 in the first two dimensions of a principal component analysis.

3.2 Analysis

Hierarchical clustering, specifically using an agglomerative method with Ward's method as described in Section 2.1.1 is applied to the Euclidean distance matrix calculated from the data set. This procedure creates clusters of all possible sizes based on the minimization in the total within cluster residual sum of squares. To avoid exploring too deeply into the cluster solution, an initial constraint that the improvement in mean of the residual sum of squares needs to be greater than the default value of 10% of the one-cluster residual sum of squares to limit the range of possible number of clusters to feasible ones (Section 2.3). After that, generalized cross validation (GCV) and ordinary cross validation (OCV) are computed on every candidate cluster solution. Lastly, three selection rules, namely, minimum OCV, relative slope and acceleration factor (mentioned in Section 2.4), are used to get the optimal suggested number of clusters for each data set. The whole procedure is implemented in a R function detailed in Appendix C.

3.3 Results

The suggested number of clusters for simulated data sets are summarized in Table 2. Minimum OCV does not suggest choosing one cluster. It tends to choose small dimensions, mostly two and in some cases, it chooses three. Overall, it is the most the conservative among the three rules. One reason may be because of large variability in OCV, which shows a big improvement in its value in two clusters from the one cluster result and then typically rapidly increases again. The behavior of OCV doesn't always have a clear trend and is sometimes difficult to predict.

The relative slope rule does a better job than the acceleration factor and the minimum OCV in estimating the correct number of clusters. It selects the correct size in all 50 cases in the *null model*, the *3-cluster model* and the *2-elongated clusters*. In the *random 4-cluster* settings (3rd and 4th settings), the performance is not as good as other settings with 37/50 correct cases for 3 dimensions and 23/50 correct cases for 10 dimensions. These are the most difficult settings as the clusters are not well–separated. The rest of the test cases were estimated with one dimension lower than the correct one and there is one case in each setting that was estimated with only 2 clusters. One of the reasons for this may be the distance between clusters are not so large. Although one may think increasing the dimensions would help gather more information for the discrepancy between clusters (Milligan & Cooper, 1985), it actually does not help in this setting because the minimum distance between clusters is still kept as 2 although they have more dimensions, so the overall distance was decreasing.

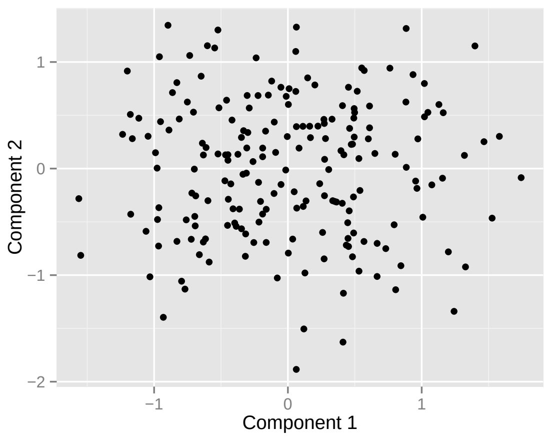
The acceleration factor has a large drawback because this measure can only be calculated when it has at least three available cross-validation results. Hence, the acceleration factor fails to detect correct one cluster results in the *null model* setting. Apart from that, this method tends to be conservative when the number of chosen clusters is much smaller than the correct results. This can be explained by realizing that GCV for hierarchical clusters decreases rapidly for the first values and then the decreases slow down gradually after that.

3.4 Results for Wheat data

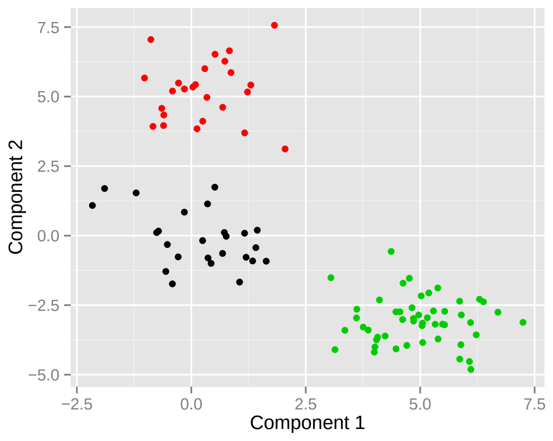
After looking at the simulated data set to see how cross validation criteria and selection rules perform, we come back to the research question raised in Section 1: (1) How many groups should the seeds be classified into, and (2) With that number of groups, which observations belong to the groups and (3) What are the characteristics of each group.

Three rules give three different answers for this data set (Fig. 7). OCV has its minimum value at 4, while the acceleration factor decides that the biggest change in slope happens at 2 clusters, and the 10% threshold of slope suggests stopping at 2. Although from the simulation study, the acceleration factor seems to be conservative, it picks the largest number of clusters here.

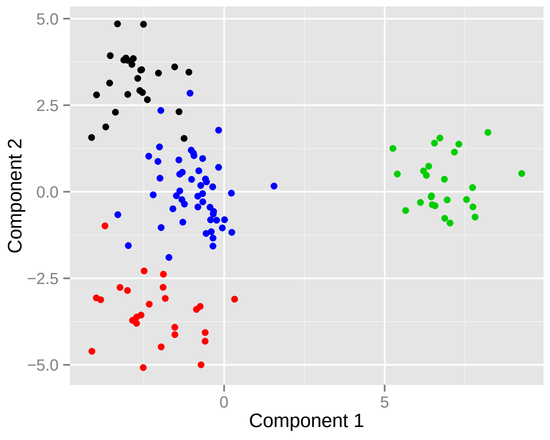
It should be noted that in the Wheat kernel data set, the information about which type of wheat each kernel came from was available. However, even though there are three different types of kernels, we do not know if the characteristics of kernels are different from type to type, or that kernels from the same type are assured to be similar to each other. That is why it is worthy to consider using cluster analysis in this situation.



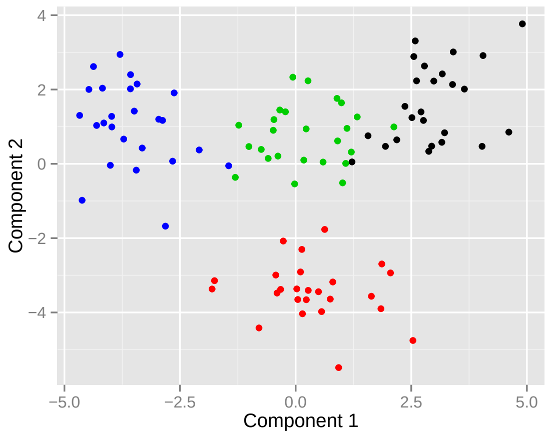
(a) Uniform in 10 dimensions



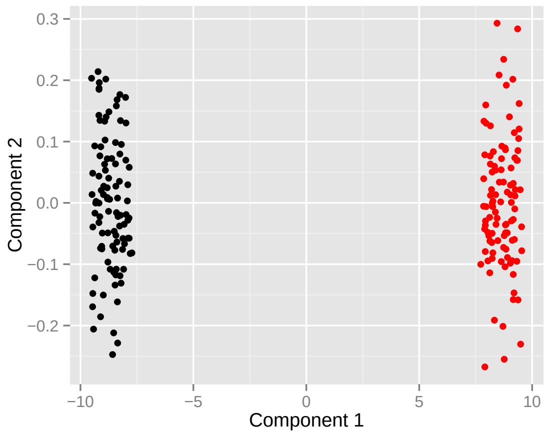
(b) Three clusters in 2 dimensions



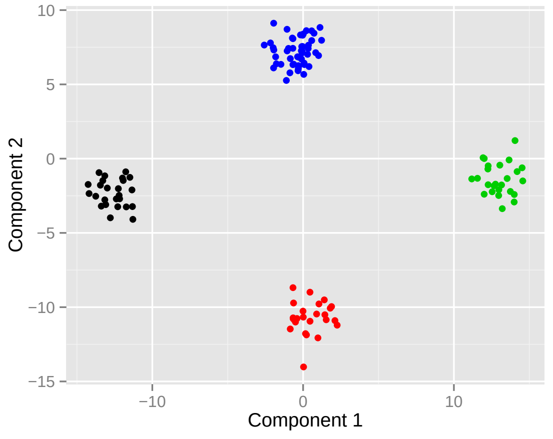
(c) Four clusters in 3 dimensions (minimum distance 2)



(d) Four clusters in 10 dimensions (minimum distance 2)



(e) Two elongated clusters in 3 dimensions



(f) Four well-separated clusters in 3 dimensions

Figure 6: The designed clusters can be seen in different simulation schemes. All data points are projected onto the first two dimensions of principal component analysis for these plots.

Table 2: Estimated number of clusters by different methods

Setting	Estimates of number of clusters										
	1	2	3	4	5	6	7	8	9	10	> 10
Null model in 10 dimensions											
Min OCV	0	37	4	5	2	2	0	0	0	0	0
Relative slope	50	0	0	0	0	0	0	0	0	0	0
Acceleration Factor	NA[a]	27	16	4	1	1	0	1	0	0	0
3-cluster model											
Min OCV	0	50	0	0	0	0	0	0	0	0	0
Relative slope	0	0	50	0	0	0	0	0	0	0	0
Acceleration Factor	NA[a]	50	0	0	0	0	0	0	0	0	0
Random 4-cluster model in 3 dimensions											
Min OCV	0	38	7	3	2	0	0	0	0	0	1
Relative slope	0	0	3	47	0	0	0	0	0	0	0
Acceleration Factor	NA[a]	21	11	18	0	0	0	0	0	0	0
Random 4-cluster model in 10 dimensions											
Min OCV	0	32	9	5	2	2	0	1	0	0	0
Relative slope	0	0	26	24	0	0	0	0	0	0	0
Acceleration Factor	NA[a]	21	11	18	0	0	0	0	0	0	0
2 elongated clusters[b]											
Min OCV	0	50	0	0	0	0	0	0	0	0	0
Relative slope	0	50	0	0	0	0	0	0	0	0	0
Acceleration Factor	NA[a]	50	0	0	0	0	0	0	0	0	0
Four well separated clusters in 3 dimensions[b]											
Min OCV	0	26	24	0	0	0	0	0	0	0	0
Relative slope	0	0	0	50	0	0	0	0	0	0	0
Acceleration Factor	NA[a]	22	0	28	0	0	0	0	0	0	0

[a] due to the formula, the smallest number of clusters that acceleration factor cannot be 1.

[b] because the distance between two clusters are two large, `cp` is set to be 0.001 in this setting in order for af to work.

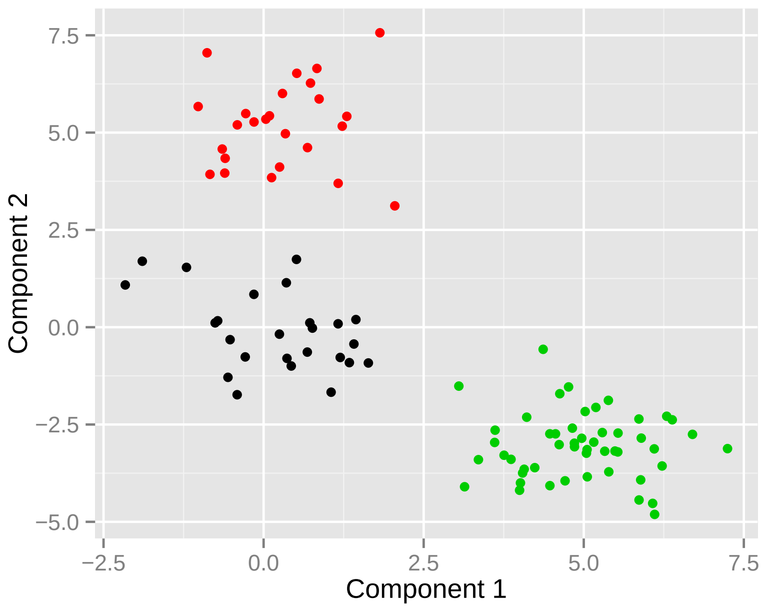


Figure 7: Two-dimensional representation of the three cluster solution by three CV-based selection rules.

4 Conclusions & Future Work

Cross-validation has been used widely in many statistical methods and has been proved to be effective for the model refinement and tuning, especially when trying to optimize the predictive ability of the methods. Nevertheless, as far as we know, the application of cross-validation into clustering has not been formally explored. We proposed three different rules for using CV-related measures for choosing the number of clusters in Ward's hierarchical clustering, and performed a simulation study with different scenarios to see how well cross-validation criteria and selection rules perform. The results we've got so far are moderately promising for hierarchical clustering method with Ward's method. But these three cross-validation based selection rules tend to be conservative, they usually choose a smaller number of clusters than the truth.

There are many potential extensions that can be done in the future to further explore the potential applications of these methods.

- Try other selection rules used in similar situations. The one standard error rule in C&RT (De'ath & Fabricius, 2000) could be one of the candidates, although this will only make the results more conservative.
- Apply cross-validation to other clustering methods, such as k-means clustering, monothetic clustering, etc. where it might provide less conservative results.
- Use other linkage and distance functions to see if the performance of cross-validation will change.
- Implement cross-validation for selecting number of clusters into an R package and publish for wider consideration.

A Appendix 1: Proof of Residual Sum of Square Connection to Distance Matrix

$$
\begin{aligned}
\sum_{j=1}^{p}\sum_{i=1}^{n}(x_{ij}-\bar{x_j})^2 &= \sum_{j=1}^{p}\sum_{i=1}^{n}\left(x_{ij}-\frac{\sum_{k=1}^{n}x_{kj}}{n}\right)^2 \\
&= \frac{1}{n^2}\sum_{j=1}^{p}\sum_{i=1}^{n}\left(nx_{ij}-\sum_{k=1}^{n}x_{kj}\right)^2 \\
&= \frac{1}{n^2}\sum_{j=1}^{p}\sum_{i=1}^{n}\left(\sum_{k=1}^{n}(x_{ij}-x_{kj})\right)^2 \\
&= \frac{1}{n^2}\sum_{j=1}^{p}\sum_{i=1}^{n}\left(\sum_{k=1}^{n}(x_{ij}-x_{kj})^2+2\sum_{l}\sum_{m<l}(x_{ij}-x_{lj})(x_{ij}-x_{mj})\right) \\
&= \frac{1}{n^2}\sum_{j=1}^{p}\sum_{i=1}^{n}\left(\sum_{k=1}^{n}(x_{ij}-x_{kj})^2+2\sum_{l}\sum_{m<l}(x_{ij}^2-x_{ij}x_{lj}-x_{ij}x_{mj}+x_{lj}x_{mj})\right) \\
&= \frac{1}{n^2}\sum_{j=1}^{p}\sum_{i=1}^{n}\left(\sum_{k=1}^{n}(x_{ij}-x_{kj})^2+\sum_{l}\sum_{m<l}(x_{ij}-x_{lj})^2+(x_{ij}-x_{mj})^2-(x_{lj}-x_{mj})^2\right) \\
&= \frac{1}{n^2}\sum_{j=1}^{p}\sum_{i=1}^{n}\left(\sum_{k=1}^{n}(x_{ij}-x_{kj})^2+(n-1)\sum_{k=1}^{n}(x_{ij}-x_{kj})^2-\sum_{l}\sum_{m<l}(x_{lj}-x_{mj})^2\right) \\
&= \frac{1}{n^2}\sum_{j=1}^{p}\sum_{i=1}^{n}\left(n\sum_{k=1}^{n}(x_{ij}-x_{kj})^2-\sum_{l}\sum_{m<l}(x_{lj}-x_{mj})^2\right) \\
&= \frac{1}{n^2}\sum_{i=1}^{n}\left(n\sum_{j=1}^{p}\sum_{k=1}^{n}(x_{ij}-x_{kj})^2-\sum_{j=1}^{p}\sum_{l}\sum_{m<l}(x_{lj}-x_{mj})^2\right) \\
&= \frac{1}{n^2}\sum_{i=1}^{n}\left(nd_{i\bullet}^2-A\right) \\
&= \frac{1}{n^2}\left(2nA-nA\right)=\frac{A}{n}
\end{aligned}
$$

B Appendix 2: Proof of Prediction Error Connection to Distance Matrix

Let the response matrix be X with size $n \times p$. A new observation has its response $x_{*\cdot}$.

$$\begin{aligned}
\sum_{j=1}^{p}(x_{*j} - \bar{x_j})^2 &= \sum_j \left(x_{*j} - \frac{\sum_{i=1}^n x_{ij}}{n} \right)^2 \\
&= \frac{1}{n^2} \sum_j \left(n x_{*j} - \sum_i x_{ij} \right)^2 \\
&= \frac{1}{n^2} \sum_j \left(\sum_i (x_{*j} - x_{ij}) \right)^2 \\
&= \frac{1}{n^2} \sum_j \left(\sum_i (x_{*j} - x_{ij})^2 + 2 \sum_k \sum_{l<k} (x_{*j} - x_{lj})(x_{*j} - x_{kj}) \right) \\
&= \frac{1}{n^2} \left(\sum_j \sum_i (x_{*j} - x_{ij})^2 + 2 \sum_j \sum_k \sum_{l<k} (x_{*j} - x_{lj})(x_{*j} - x_{kj}) \right)
\end{aligned}$$

Let d_{*i}^2 be the squared Euclidean distance between site $*$ and site i, then
$d_{*i}^2 = \sum_j (x_{*j} - x_{ij})^2$
and A is the sum squared distance between $*$ and other observations, then $A =$

$\sum_i d_{*i}^2$

$$
\begin{aligned}
\sum_{j=1}^{p}(x_{*j}-\bar{x_j})^2 &= \frac{1}{n^2}\left(A+2\sum_j\sum_k\sum_{l<k}(x_{*j}-x_{lj})(x_{*j}-x_{kj})\right)\\
&= \frac{1}{n^2}\left(A+2\sum_j\sum_k\sum_{l<k}(x_{*j}^2-x_{*j}x_{kj}-x_{*j}x_{lj}+x_{kj}x_{lj})\right)\\
&= \frac{1}{n^2}\left(A+\sum_j\sum_k\sum_{l<k}2(x_{*j}^2-x_{*j}x_{kj}-x_{*j}x_{lj}+x_{kj}x_{lj})\right)\\
&= \frac{1}{n^2}\left(A+\sum_j\sum_k\sum_{l<k}\left[(x_{*j}-x_{kj})^2+(x_{*j}-x_{lj})^2-(x_{kj}-x_{lj})^2\right]\right)\\
&= \frac{1}{n^2}\left(A+\sum_j\sum_k\sum_{l<k}(x_{*j}-x_{kj})^2+\sum_j\sum_k\sum_{l<k}(x_{*j}-x_{lj})^2-\sum_j\sum_k\sum_{l<k}(x_{kj}-x_{lj})^2\right)\\
&= \frac{1}{n^2}\left(A+(n-1)\sum_j\sum_{i=1}^{n}(x_{*j}-x_{ij})^2-B\right)\\
&= \frac{1}{n^2}\left(A+(n-1)A-B\right)\\
&= \frac{A}{n}-\frac{B}{n^2}
\end{aligned}
$$

with $B=\sum_k\sum_{l<k}\sum_j(x_{kj}-x_{lj})^2=\sum_k\sum_{l<k}d_{kl}^2$

C Appendix 3: Cluster.cv.R function

```
cluster.cv <- function(data, dtype = "dist", method="ward.D", cp = 0.01,
    improvement = 0.1) {

    # Check if the input data is distance matrix or data matrix,
    # then create Euclidean distance matrix and squared E. distance matrix
    if (dtype == "dist") {
        dist <- data
        dist2 <- data^2
```

```
} else if (dtype == "data.frame") {
    dist <- dist(data)
    dist2 <- dist(data)^2
} else
    stop("Wrong data type. Must be either result of dist or data.frame.")

# Initialize local variables
n <- attributes(dist)$Size # Number of variables
data.m <- as.matrix(dist2) # Type conversion for model fitting
ori.sse <- sum(dist2)/n # RSS

# Initial rule: improvement of RSS
alpha <- cp * ori.sse

# Hierarchical clustering by the given method (default: Ward's method)
hca <- hclust(dist, method)

gcv <- rep(NA, n)
ocv <- rep(NA, n)
mgcv <- rep(NA, n)
rss <- rep(NA, n)
i <- 1
improv1 <- ori.sse

while ((i <= n) && (improv1 > alpha)) {
    memb <- cutree(hca, k=i)
    # Calculate total RSS for each cluster structure
    if (i == 1)
        rss[i] <- sum(dist2)/n
    else {
        rss[i] <- 0
        for (j in 1:i) {
            clus <- data.m[which(memb==j),which(memb==j)]
            rss[i] <- rss[i] + sum(clus) / (2*sum(memb==j))
        }
    }

    # GCV
    trA <- i
    gcv[i] <- n * rss[i] / (n-trA) ^2
    improv1 <- rss[i]
    # OCV
    ocv[i] <- 0
    if (dtype=="data.frame") {
        mdata <- as.matrix(data)
        model <- lm(mdata ~ memb)
        fit <- fitted(model)
        for (j in 1:n) {
            enum <- sum((mdata[j,] - fit[j,])^2)
            Aii <- influence(model)$hat[j]
            denom <- (1-Aii)^2
            ocv[i] <- ocv[i] + (1/n)*(enum / denom)
        }
    }

    i <- i + 1
}

# Relative slope rule
j <- 2
improv2 <- 1
# While the relative slope is still greater than "percentage of improvement"
while ((improv2 >= improvement) && (j < i)) {
    improv2 <- (gcv[j-1] - gcv[j]) / gcv[1]
    j <- j + 1
}
if (j == i) gcv1 <- gcv[1:(j-1)] else
    gcv1 <- gcv[1:(j-2)]
nclusters <- which(gcv1 == min(gcv1))

# Min OCV rule
ocv <- ocv[!is.na(ocv)]
nclusters.ocv <- which(ocv == min(ocv))

# Acceleration factor rule
if ((i-1) >= 3) {
    acceleration <- rep(-9999, i-2)
    for (j in 2:(i-2)) {
        acceleration[j] <- gcv[j+1] - 2*gcv[j] + gcv[j-1]
        if (gcv[j+1] - gcv[j] > 0) break
    }
```

```
        nclusters.acc <- which(acceleration == max(acceleration)) - 1
    } else nclusters.acc <- NA

    # Return data
    ret1 <- list(rss=rss[!is.na(rss)], gcv=gcv[!is.na(gcv)],
                 ocv=ocv,
                 nclusters=nclusters,
                 nclusters.acc = nclusters.acc,
                 nclusters.ocv = nclusters.ocv)
    class(ret1) <- "cluster.cv"
    ret1
}
```

References

Bache, Kevin, & Lichman, Mosche. 2013. *seeds Data Set – UCI Machine Learning Repository.*

Craven, Peter, & Wahba, Grace. 1978. Smoothing noisy data with spline functions. *Numerische Mathematik*, **31**(4), 377–403.

De'ath, Glenn. 2002. Multivariate Regression Tree: A New Technique for Modeling Species–Environment Relationships. *Ecology*, **83**(4), 1105–1117.

De'ath, Glenn, & Fabricius, Katharina E. 2000. Classification and Regression Trees: A Powerful Yet Simple Technique for Ecological Data Analysis. *Ecology*, **81**(11), 3178–3192.

Everitt, Brian, & Hothorn, Torsten. 2011. *An Introduction to Applied Multivariate Analysis with R.* 1 edn. Springer.

Everitt, Brian S., Landau, Sabine, Leese, Morven, & Stahl, Daniel. 2011. *Cluster Analysis.* 5 edn. Wiley.

Greenwood, Mark. 2014. *Lecture Notes for Multivariate Data Analysis Course in Spring 2014.* Montana State University.

James, Gareth, Witten, Daniela, Hasite, Trevor, & Tibshirani, Robert. 2013. *An Introduction to Statistical Learning: with Applications in R.* 1st edn. Springer.

Milligan, Glenn W., & Cooper, Martha C. 1985. An examination of procedures for determining the number of clusters in a data set. *Psychometrika*, **50**(2), 159–179.

Raîche, Gilles, Walls, Theodore A., Magis, David, Riopel, Martin, & Blais, Jean-Guy. 2013. Non-Graphical Solutions for Cattell's Scree Test. *Methodology: European Journal of Research Methods for the Behavioral and Social Sciences*, **9**(1), 23–29.

Robison-Cox, Jim. 2012. *Lecture Notes for Linear Models Course in Fall 2012*. Montana State University.

Therneau, Terry, Atkinson, Beth, & Ripley, Brian. 2014. *rpart: Recursive Partitioning and Regression Trees*. R package version 4.1-8.

Tibshirani, Robert, Walther, Guenther, & Hastie, Trevor. 2001. Estimating the number of clusters in a data set via the gap statistic. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, **63**(2), 411–423.

Walesiak, Marek, & Dudek, Andrzej. 2014. *clusterSim: Searching for optimal clustering procedure for a data set*. R package version 0.43-4.

Ward, Joe H. 1963. Hierarchical Grouping to Optimize an Objective Function. *Journal of the American Statistical Association*, **58**(301), 236–244.

Wood, Simon. 2006. *Generalized Additive Models: An Introduction with R*. 1 edn. Chapman and Hall/CRC.