clues: An R Package for Nonparametric Clustering Based on Local Shrinking

Fang Chang
York University

Weiliang Qiu
Harvard Medical School

Ruben H. Zamar
University of British Columbia

Ross Lazarus
Harvard Medical School

Xiaogang Wang
York University

Abstract

This introduction to the R package **clues** is a (slightly) modified version of Chang *et al.* (2010), published in the *Journal of Statistical Software*.

Determining the optimal number of clusters appears to be a persistent and controversial issue in cluster analysis. Most existing R packages targeting clustering require the user to specify the number of clusters in advance. However, if this subjectively chosen number is far from optimal, clustering may produce seriously misleading results. In order to address this vexing problem, we develop the R package **clues** to automate and evaluate the selection of an optimal number of clusters, which is widely applicable in the field of clustering analysis. Package **clues** uses two main procedures, shrinking and partitioning, to estimate an optimal number of clusters by maximizing an index function, either the CH index or the Silhouette index, rather than relying on guessing a pre-specified number. Five agreement indices (Rand index, Hubert and Arabie's adjusted Rand index, Morey and Agresti's adjusted Rand index, Fowlkes and Mallows index and Jaccard index), which measure the degree of agreement between any two partitions, are also provided in **clues**. In addition to numerical evidence, **clues** also supplies a deeper insight into the partitioning process with trajectory plots.

Keywords: agreement index, cluster analysis, dissimilarity measure, K-nearest neighbor.

1. Introduction

Cluster analysis, an organization of a collection of patterns into clusters based on selected similarity (or dissimilarity) measures, is an unsupervised technique which is widely applied by researchers from different disciplines. In dealing with real problems, decision makers should make as few assumptions about the data set as possible, because of the limited availability of prior information. This is a mature area in decision theory (Jain *et al.* 1999). Currently, K-means (MacQueen 1967; Hartigan and Wong 1979) is one of the most popularly adopted partitioning algorithms, as evidenced by its use in a wide variety of packages in the R system for statistical computing (R Development Core Team 2009), such as **cclust** (Dimtriadou 2009), **clue** (Hornik 2005, 2009), **clustTool** (Templ 2008), among others. Another partitioning algorithm, PAM (Kaufman and Rousseeuw 1990), which is considered as a more robust version of K-means by partitioning of the data into K clusters around medoids, is increasingly popular

as seen in **cluster** (Maechler *et al.* 2009) and **fpc** (Hennig 2007). In addition to these partitioning clustering algorithms, an alternative approach, hierarchical clustering, is commonly used for microarray data (Chipman and Tibshirani 2006). R packages **hybirdHclust** (Chipman *et al.* 2008), **segclust** (Picard *et al.* 2007; Picard 2010) and **mclust** (Fraley and Raftery 2002, 2008) use hierarchical clustering. The first takes advantage of both bottom-up clustering and top-down approaches, while the other two make use of probability-model-based hierarchical clustering.

Even though more and more up to date R packages are created targeting different clustering approaches, the optimal identification of number of clusters remains relatively less developed. Among the packages mentioned above, some criteria have been proposed to identify an optimal number of clusters. For example, **fpc** chooses a figure which corresponds to the optimal average silhouette width criterion, **mclust** selects a number that coheres to the optimal BIC value and **segclust** picks a digit that is either in accordance with an optimal modified BIC (Zhang and Siegumund 2007) or the best sequential choice of Pselect and Kselect (Picard *et al.* 2007).

In this paper, we present a novel R package **clues**, which aims to simultaneously provide an estimate of the number of clusters and to obtain a partition of a data set via local shrinking. The shrinking procedure in **clues** is an implementation of the mean-shift algorithm, but is influenced by the K-nearest neighbor approach (Mack and Rosenblatt 1979) rather than kernel functions. For the value of K, we start with a small number and gradually increase it until the preferred measure of strength, CH index (Calinski and Harabasz 1974) or Silhouette index (Kaufman and Rousseeuw 1990), is optimized (Wang *et al.* 2007). One important benefit of **clues** is its ability to identify and deal with irregular elements. In order to help validate the quality of the number of clusters and the clustering algorithm, five agreement indices are available to support decision making.

This package is available from the Comprehensive R Archive Network at `http://CRAN.R-project.org/package=clues` and installable in the R version 2.8.1 or above. The package **clues** can be installed in the usual ways and is ready to use in an R session after typing

```
R> library("clues")
```

The layout for the rest of paper is as follows. Section 2 consists of a succinct statement of the algorithm. Section 3 describes and illustrates the strength measures, dissimilarity measures, and agreement indices. Section 4 depicts the utilization of clues functions not only numerically but also graphically. A brief discussion follows in Section 5.

2. Brief algorithm

The rationale for the clues algorithm is to regard data points as particles in a gravitational field, initially with unit mass and zero velocity. Conceptually, the local gravitational field will pull each data point into a denser region according to some assumed gravitational laws and the distribution of other data points.

The clues algorithm consists of the following three procedures:

- shrinking procedure,
- partition procedure,

- determination of the optimal K.

For the shrinking procedure, the original data set is calibrated in a way that pushes each data point towards its focal point, which serves as the cluster center or mode of the probability density function. The number K is chosen iteratively. For a given K, due to the robustness of the median, each data point is designed to move to the element wise median of the set which consists of its K nearest neighbors according to dissimilarity measures, either Euclidean distance or Pearson Correlation distance in **clues**. For this process, a user supplied stopping rule is required, beyond which excess iterations will not make a significant difference in terms of accuracy, although they will prolong the computing time. After this manipulation, the mutual gaps in the data become apparent.

For the partitioning procedure, the calibrated data obtained from shrinking are used in place of the original data set. To start the partitioning procedure, one arbitrary data point is picked and replaced by its nearest fellow point, and their distance is recorded. We restart with this fellow point and repeat the first step. It is worth while to note that the selection is without replacement, so once a data point is picked for replacement, there is no opportunity for it to be chosen again in that run. In order to separate the groups, a benchmark R, which is defined as the summation of the mean distance and 1.5 times the interquartile range(the difference between the first and third quartiles)is introduced. Denote g as the group index that is initialized to be 1 and once the distance between a data point sorted by the above procedure and its fellow point surpasses R, a new group is started by increasing g by 1.

The determination of the optimal K involves optimizing the strength measure index, either CH or Silhouette. An arbitrary factor α is introduced to improve the speed of the computation. While 0.05 is the default choice in **clues**, users can reset it according to personal preference and willingness to wait for computations to complete. We use α because the choice of K has no effect on clustering result as long as it lies in a neighborhood of the optimal K, so it is chosen to minimize additional computation that will not substantially affect the outcome. The following two schemes are adopted to strengthen the power of the algorithm.

- Initialize K to be αn and set the size of increment of K to be αn for the following iterations.
- Within each iteration, use the calibrated data from the previous loop as the new data.

The overall process does not require any input other than external stopping rules. The combination of our shrinking procedure and the partitioning procedure allow the identification of an optimized K compatible with our assumptions.

3. Dissimilarity measures, agreement and strength indices

Two common dissimilarity measures, "Euclidean" and "1−Pearson Correlation", are widely used. The range for "Euclidean" distance is from 0 to infinity, and the larger the value, the further apart the two points. The range for "1−Pearson Correlation" distance is from 0 to 2, and correlation is used as a benchmark here. Two variables with positive correlation 1, have no dissimilarity, while the larger the value, the more negative correlation between the variables. These measures are of great importance through the whole procedure since the neighbors of a

data point are connected and recognized by their dissimilarity. To our knowledge, only **clues** and **hybridHclust** have implementations for both dissimilarity measures.

It is important to know if a partition obtained by a clustering algorithm is good or not. One can compare the partition with other partitions obtained by other clustering algorithms. The more agreement among these partitions, the more confidence we have on the obtained partition. Lack of agreement among these partitions indicates either the cluster structure is difficult to detect or some clustering algorithms do not perform properly for this specific data set. In this situation, a strength index that measures the separation among clusters of the data set will be useful to indicate which partition is better for this specific data set. The larger the minimum separation a partition has, the better the partition is.

Ideally, the larger a strength index is, the better the partition is. However, we should be cautious that strength indices proposed in the literature could not work as expected for all types of data. We give such a counter-example in Section 4.2. Also some strength indices (such CH index mentioned below) are informative only when used to compare two or more clusterings for the same data set since they do not have finite upper bound.

The **clues** package implements five agreement indices (Rand index, Hubert and Arabie's adjusted Rand index, Morey and Agresti's adjusted Rand index, Fowlkes and Mallows index, and Jaccard index) and two strength indices (CH index and averaged Silhouette index).

These five agreement indices were recommended by Milligan and Cooper (1986), and each one takes values between 0 and 1. They indicate how closely two partitions of the same data set match each other. The closer to 1 the value, the more similarity between the two partitions, where 1 means a perfect match.

The Silhouette index measures the average dissimilarity between a data point in a cluster to all the data points in the nearest neighboring cluster of this data point. The range of the Silhouette index is from -1 to 1. The larger the Silhouette index, the more evidence we have that the data point does not belong to its nearest neighboring cluster. The average Silhouette is the average Silhouette index of all the data points. Therefore the range of average Silhouette is also -1 to 1, which is identical to that of Silhouette index. It is a cluster validity index which reflects the compactness of the clusters and indicates whether a cluster structure is well separated or not. The higher the average Silhouette index, the better the cluster separation.

The CH index is the ratio of between cluster variation to within cluster variation and takes values between 0 and ∞.

4. Data analysis using clues

In this section, we use two real data sets from two different disciplines to show how **clues** works.

4.1. Main function: `clues`

In R, a typical call for using `clues` is

```
clues(y, strengthMethod = "sil", disMethod = "Euclidean")
```

The function documentation regarding explicit instruction on input arguments is given online by the command `help(clues)`.

A data matrix y is the only compulsory input, and all the other arguments are optional which makes `clues` more convenient to use. Note that compared to other nonhierarchical clustering packages, `clues` avoids the need for the input of the number of clusters to be subjectively assigned.

As an initial illustration, we consider the "Wisconsin Diagnostic Breast Cancer" data set (`WDBC`, Asuncion and Newman 2007) with 569 samples (rows) and 32 variables (columns). The first column shows ID number, the second column is the diagnosis result which is either "benign" or "malignant". And it turns out to be 357 benign and 212 malignant with no missing values. All the rest columns are real-valued numbers with each one standing for certain index. The data set along with its detailed illustration on the source information is available from UCL Machine Learning Repository (Asuncion and Newman 2007). We take a first glance of the first 6 columns of the data:

```
R> WDBC <- read.table("WDBC.dat", sep = ",", header = FALSE)
R> colnames(WDBC) <- c("ID", "caco", paste("dim", 1:30, sep = ""))
R> WDBC.mem <- rep(NA, length(WDBC$caco))
R> WDBC.mem[WDBC$caco == "M"] <- 1
R> WDBC.mem[WDBC$caco == "B"] <- 2
R> head(WDBC[, 1:6])

        ID caco  dim1  dim2   dim3   dim4
1   842302    M 17.99 10.38 122.80 1001.0
2   842517    M 20.57 17.77 132.90 1326.0
3 84300903    M 19.69 21.25 130.00 1203.0
4 84348301    M 11.42 20.38  77.58  386.1
5 84358402    M 20.29 14.34 135.10 1297.0
6   843786    M 12.45 15.70  82.57  477.1
```

The variations of the 30 variables (not include ID and *caco*) are quite different.

```
R> sdVec <- apply(WDBC[, -c(1:2)], 2, sd)
R> round(quantile(sdVec), 3)

     0%     25%     50%     75%    100%
  0.003   0.019   0.073   4.107 569.357
```

Hence, we standardize each of these variables by subtracting its sample mean and then dividing its sample standard deviation before applying function `clues` on `WDBC` with the default options.

```
R> dat <- as.matrix(WDBC[, -c(1:2)])
R> dat.s <- apply(dat, 2, scale, center = TRUE, scale = TRUE)
R> colnames(dat.s) <- colnames(WDBC)[-c(1:2)]
R> res.sil <- clues(dat.s, strengthMethod = "sil", disMethod = "Euclidean")
```

We can get a summary of the clustering results obtained by `clues` via the `summary` method:

```
R> summary(res.sil)
```

```
Number of data points:
[1] 569

Number of variables:
[1] 30

Number of clusters:
[1] 2

Cluster sizes:
[1] 186 383

Strength method:
[1] "sil"

avg Silhouette:
[1] 0.3359515

dissimilarity measurement:
[1] "Euclidean"


Available components:
 [1] "K"              "size"           "mem"            "g"
 [5] "avg.s"          "s"              "K.vec"          "g.vec"
 [9] "myupdate"       "y.old1"         "y.old2"         "y"
[13] "strengthMethod" "disMethod"
```

The number of clusters estimated by `clues` is 2 which is exactly the same as the true number of clusters.

We also can visualize the clustering results by the `plot` method, plot(obj), where `obj` is the object returned by the function `clues`. The `plot` method combines the functions `plotClusters` and `plotAvgCurves`. A simple menu will prompt the user to choose to produce either scatter plot, or plot of average trajectories, or both.

The following R code will produce scatter plot (Figure 1) for four arbitrarily chosen dimensions (1, 4, 9, and 10):

```
R> plotClusters(dat.s, res.sil$mem, plot.dim = c(1, 4, 9, 10), cex.points = 0.01)
```

The following R code will produce average trajectory plots (Figure 2) for the two clusters:

```
R> plotAvgCurves(dat.s, res.sil$mem, cex.axis = 0.3, ylim = c(-1,
+     1.5), las = 1, cex.axis = 1)
```

Both the scatter plot and the plot of average trajectories show the two clusters are not separated and are difficult to detect. The low value of the average Silhouette index also confirms this:

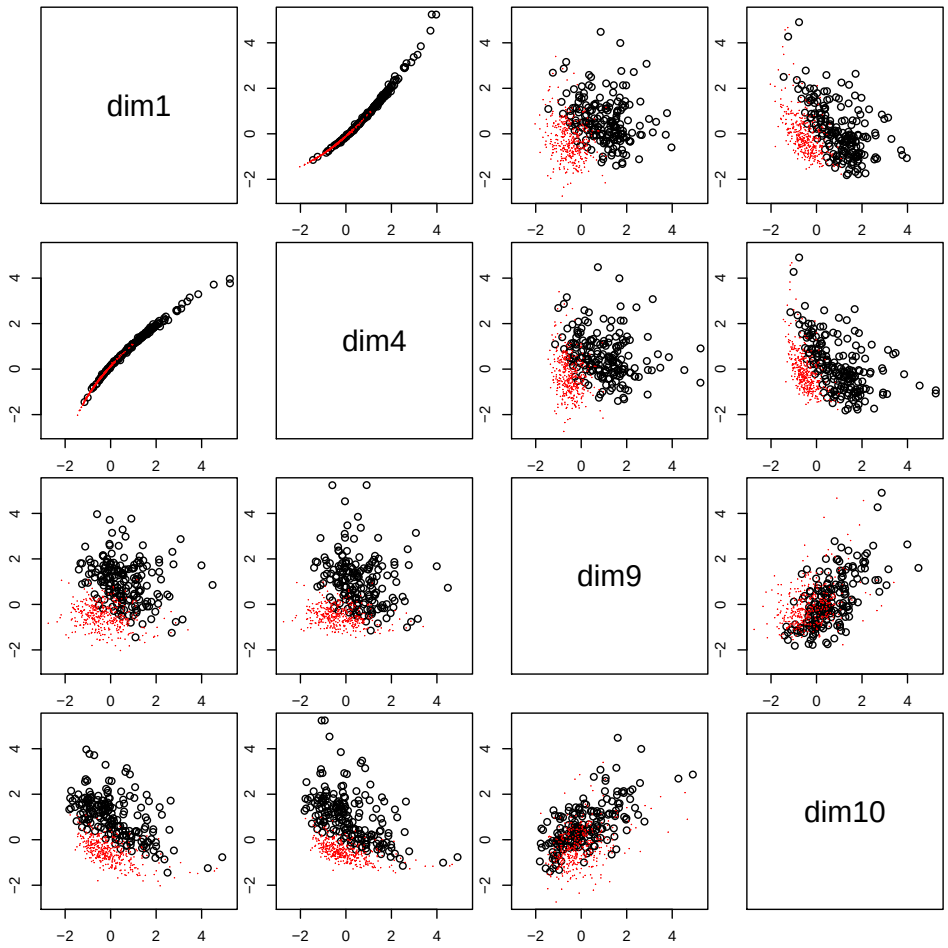

Figure 1: Pairwise scatter plots of four arbitarily chosen dimensions. The two clusters obtained by `clues` are indicated by different symbols and colors.

```
R> get_Silhouette(dat.s, res.sil$mem)$avg.s

[1] 0.3359515
```

Now that the exact partition (indicated by the variable `caco`) is already given, the accuracy of our method can be easily computed and it is listed below with those of `kmeans` and `pam`. The number of clusters required as input for `kmeans` and `pam` is specified as the true number (2) of clusters.

```
R> accuracy <- sum(WDBC.mem == res.sil$mem)/length(WDBC.mem)
R> round(accuracy, 2)

[1] 0.94

R> library("cluster")
R> set.seed(12345)
```

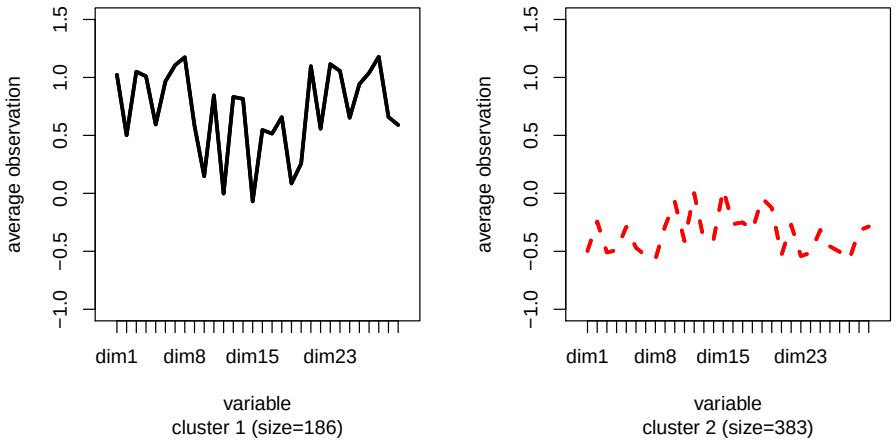

Figure 2: Plots of average trajectories for the 2 clusters obtained by `clues` of the WDBC data set

```
R> res.km <- kmeans(dat.s, 2, algorithm = "MacQueen")
R> res.pam <- pam(dat.s, 2)
R> accuracy.km <- sum(WDBC.mem == res.km$cluster)/length(WDBC.mem)
R> round(accuracy.km, 2)

[1] 0.09

R> accuracy.pam <- sum(WDBC.mem == res.pam$clustering)/length(WDBC.mem)
R> round(accuracy.pam, 2)

[1] 0.89
```

Considering that the two clusters are not separated, the accuracy 0.94 of `clues` is acceptable in this context compared to those of `kmeans` and `pam`.

4.2. Validation function: `compClust`

To view agreements among the partitions derived from different clustering methods, we can use the function `compClust`, which calculates the values of five agreement indices mentioned in Section 3 for any pair of the clustering methods considered. The syntax of `compClust` is listed below:

```
compClust(y, memMat, disMethod = "Euclidean")
```

with

- `y`: input data in matrix or data frame form.
- `memMat`: cluster membership matrix with each column represents one partition.

- `disMethod`: dissimilarity measure, either `"Euclidean"` or `"1-corr"`.

The numeric output of this function also includes strength indices which consist of CH index and average Silhouette index.

For `WDBC` data, we compared `clues` with both CH index and Silhouette index, `kmeans`, and `pam` by using `compClust`. The clustering obtained from `clues` with CH is exactly the same as that obtained from `clues` with Silhouette index. Moreover, it is closer to the true partition than those obtained from `kmeans` and `pam` in terms of the five agreement indices the degree of agreement between the partition obtained by `clues` with both CH index and Silhouette index and the true partition is 0.89 in terms of Fowlkes and Mallows index, the degree reduced to 0.84 when we compare the partition obtained by kmeans with the true partition, and the degree reduced to 0.82 when we compare the partition obtained by pam with the true partition.

```
R> res.CH <- clues(dat.s, strengthMethod = "CH", quiet = TRUE)
R> memMat <- cbind(WDBC.mem, res.sil$mem, res.CH$mem, res.km$cluster,
+     res.pam$clustering)
R> colnames(memMat) <- c("true", "clues.sil", "clues.CH", "km",
+     "pam")
R> res <- compClust(dat.s, memMat)
R> print(sapply(res, round, digits = 2))

$avg.s
     true clues.sil  clues.CH        km       pam
     0.29      0.34      0.34      0.34      0.35

$CH
     true clues.sil  clues.CH        km       pam
   225.39    257.36    257.36    267.69    264.61

$Rand
          true clues.sil clues.CH   km  pam
true      1.00      0.88     0.88 0.83 0.81
clues.sil 0.88      1.00     1.00 0.92 0.90
clues.CH  0.88      1.00     1.00 0.92 0.90
km        0.83      0.92     0.92 1.00 0.95
pam       0.81      0.90     0.90 0.95 1.00

$HA
          true clues.sil clues.CH   km  pam
true      1.00      0.76     0.76 0.65 0.61
clues.sil 0.76      1.00     1.00 0.84 0.80
clues.CH  0.76      1.00     1.00 0.84 0.80
km        0.65      0.84     0.84 1.00 0.90
pam       0.61      0.80     0.80 0.90 1.00

$MA
```

```
          true clues.sil clues.CH   km  pam
true      1.00      0.76     0.76 0.65 0.61
clues.sil 0.76      1.00     1.00 0.84 0.80
clues.CH  0.76      1.00     1.00 0.84 0.80
km        0.65      0.84     0.84 1.00 0.90
pam       0.61      0.80     0.80 0.90 1.00

$FM
          true clues.sil clues.CH   km  pam
true      1.00      0.89     0.89 0.84 0.82
clues.sil 0.89      1.00     1.00 0.93 0.91
clues.CH  0.89      1.00     1.00 0.93 0.91
km        0.84      0.93     0.93 1.00 0.96
pam       0.82      0.91     0.91 0.96 1.00

$Jaccard
          true clues.sil clues.CH   km  pam
true      1.00      0.80     0.80 0.73 0.70
clues.sil 0.80      1.00     1.00 0.86 0.84
clues.CH  0.80      1.00     1.00 0.86 0.84
km        0.73      0.86     0.86 1.00 0.92
pam       0.70      0.84     0.84 0.92 1.00
```

The values of both strength indices for the true clustering are the smallest compared to the estimated clusterings obtained by clustering methods. This informs us that we need to be cautious to use the strength indices to compare different clustering methods.

4.3. Sharpening function: `shrinking`

The function `shrinking` provides a way to sharpen the boundaries among data groups. The usage of the function is shown below:

```
shrinking(y, K, disMethod = "Euclidean", eps = 1e-04, itmax = 20)
```

where `y` is the data matrix with data points as rows and variables as columns, `K` is the number of nearest neighbors, based on which coordinate-wise medians will be used to replace the old coordinates of a data point, `disMethod` indicates dissimilarity measurement, `eps` is a small positive number (a positive value below `eps` will be regarded as zero), and `itmax` indicates how many runs of data sharpening to be performed.

We use the famous `iris` data set to illustrate this function `shrinking`. The `iris` (Fisher 1936) data set is the number 1 most popular data sets among the 187 data sets maintained at UCI Machine Learning Repository (Asuncion and Newman 2007). This data set consists of measurements of sepal length and width, and petal length and width for 3 types ("setosa", "versicolor" and "virginica") of iris flowers. There are 50 flowers in each type in the `iris` data set.

Figure 3 shows the pairwise scatter plots for the original iris data, while Figure 4 shows the pairwise scatter plots for the sharpened data (`K = 60` and `itmax = 3`). After data sharpening, the three clusters are much clearer compared to the original `iris` data set.

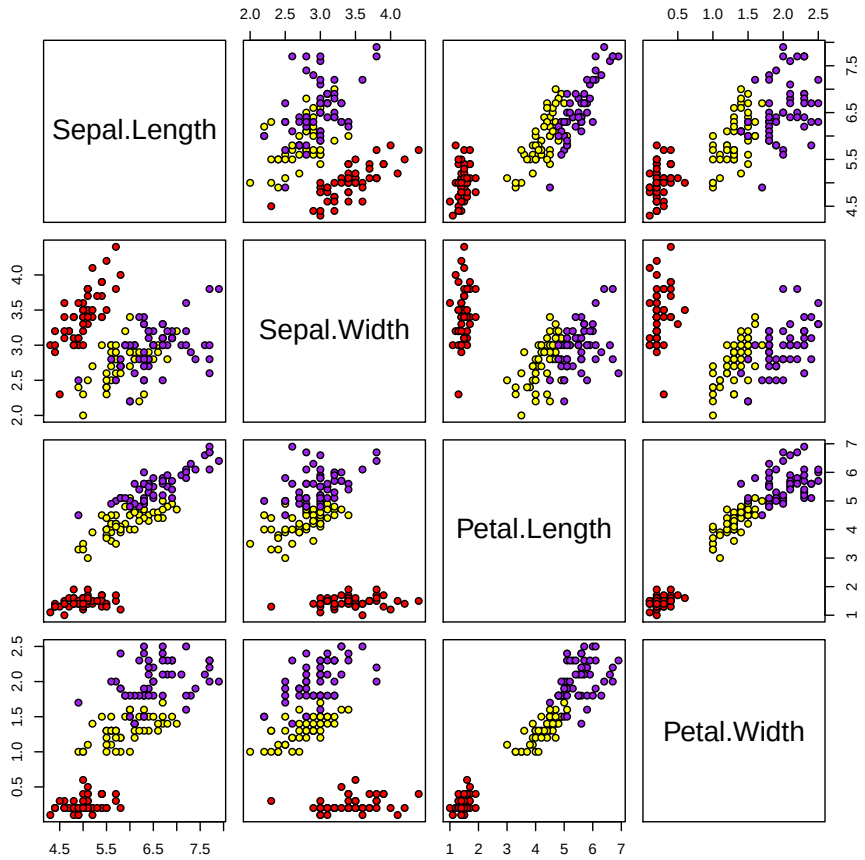

Figure 3: Pairwise scatter plots for the original iris data.

```
R> data("iris")
R> jj <- as.matrix(iris[, 1:4])
R> pairs(jj, pch = 21, bg = c("red", "yellow", "purple")[unclass(iris$Species)])
R> shrinkres <- shrinking(jj, K = 60, disMethod = "Euclidean", itmax = 3)
R> dimnames(shrinkres) <- dimnames(jj)
R> convergepnts <- shrinkres[c(1, 51, 52, 54), ]
R> row1 <- t(matrix(jitter(rep(convergepnts[1, ], 50)), nrow = 4))
R> row2 <- t(matrix(jitter(rep(convergepnts[2, ], 51)), nrow = 4))
R> row3 <- t(matrix(jitter(rep(convergepnts[3, ], 18)), nrow = 4))
R> row4 <- t(matrix(jitter(rep(convergepnts[4, ], 31)), nrow = 4))
R> shrinkres_jitter <- rbind(row1, row2, row3, row4)
R> shrinkres_group <- as.factor(rep(1:3, c(50, 51, 49)))
R> colnames(shrinkres_jitter) <- colnames(shrinkres)
R> pairs(shrinkres_jitter[, 1:4], pch = 21, bg = c("red", "purple",
+     "yellow")[unclass(shrinkres_group)])
```

The number K was picked by the authors for convenience, but can be changed by the user, and

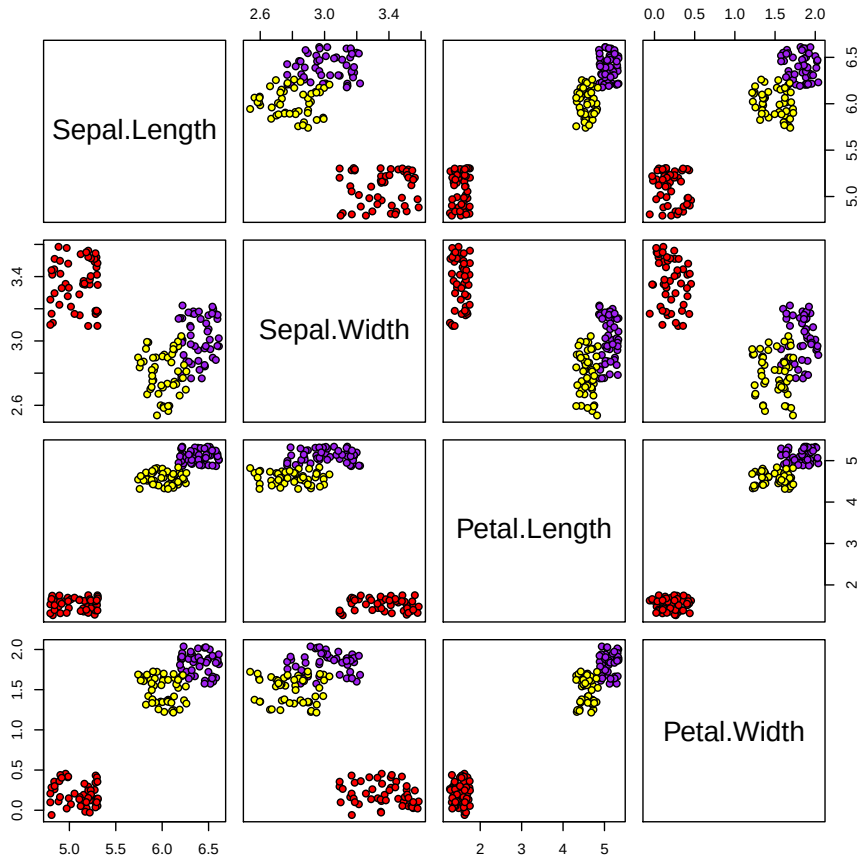

Figure 4: Pairwise scatter plot for sharpened iris data.

it is very likely that distinct values will produce variable results, because of some instability in the K nearest neighbor method. For example, if we choose a bigger $K = 90$ in `shrinking`, it will give us 2 clusters rather than 3.

5. Discussion

In this paper, we describe the methods and use of a recently developed clustering package **clues**. One major advantage of the novel **clues** algorithm is that it avoids the need of subjectively selecting a preassigned number of clusters, by iterative shrinking and clustering procedures. Secondly, it provides both numerical and graphical supports (`plotClusters`, `plotCurves` and `plotAvgCurves`) to assess the quality of the resulting partitions. Thirdly, it provides objective methods for comparing different partitioning algorithms using several popular figures of merit. Last but not least, researchers can use this package, especially functions `adjustRand` and `compClust`, to assess a proposed clustering algorithm in their simulation studies where the true partition is known.

We recommend that the users run the current version of **clues** in the background mode

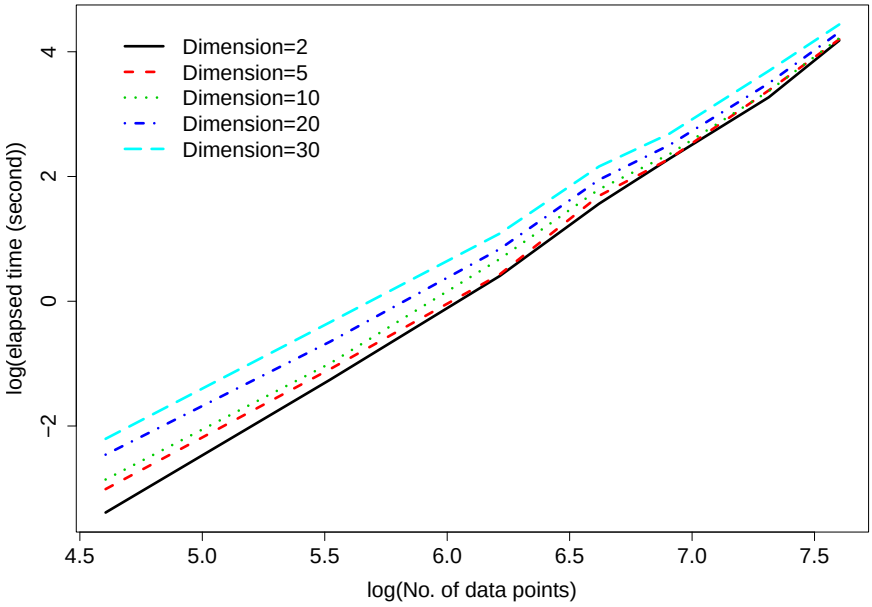

Figure 5: Plot of log average of the total elapsed times vs log number of data points in a simulation study.

for large data sets due to that the search for K nearest neighbors takes time. To get a rough idea about the relationship between running time and the number of clusters and the number of dimensions, we conducted a small simulation study. In this simulation study, each simulated data set contains two equal sized clusters generated from two multivariate normal distributions. The covariance matrices are both identity matrices. The mean vector of the first cluster is the vector of five (i.e. all elements of the mean vector is 5). The mean vector of the second cluster is the vector of zero. The numbers of data points are 100, 250, 500, 750, 1000, 1500, and 2000, respectively. The numbers of dimensions are 2, 5, 10, 20, and 30, respectively. For each combination of the number of data points and the number of dimensions, we simulated 100 data sets. We conducted this simulation study in a personal computer with Microsoft Windows XP Professional operating system. The total physical memory is $4,100$ MB; total virtual memory is 2 GB. The processor is x86 Family 6 Model 23 Stepping 7 GenuineIntel 2500 Mhz.

The plot of log average of the total elapse times versus log number of data points (Figure 5) indicates the running times scale roughly as the square of the number of data points.

The plot of log average of the total elapse times versus log number of dimensions (Figure 6) indicates the running times increase linearly as the number of dimensions increases. But the slopes are quite flat compared to those in Figure 5.

We will improve the speed of **clues** in future version by using more efficient K nearest neighbor searching algorithm.

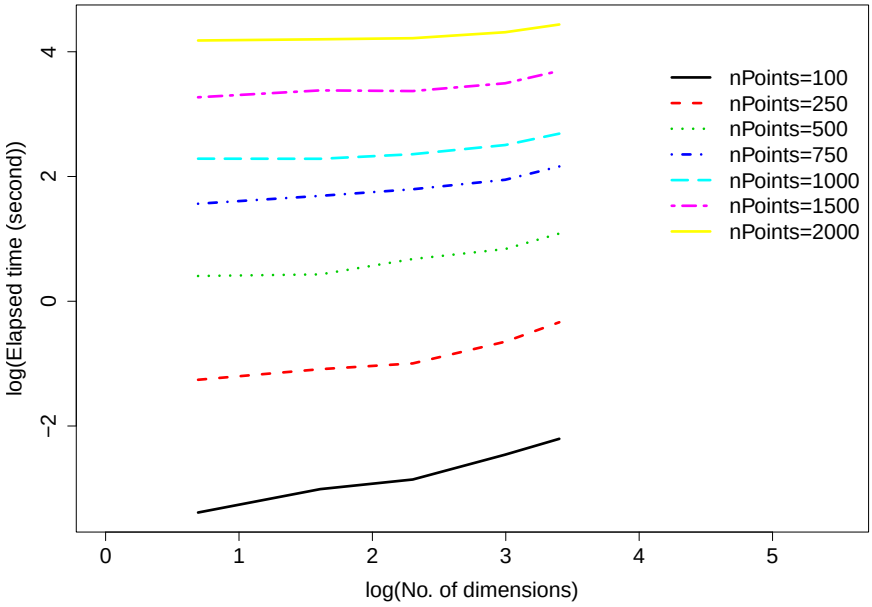

Figure 6: Plot of log average of total elapsed times versus log number of dimensions in a simulation study.

Acknowledgments

The authors would like to thank referees and Editor Jan de Leeuw for their valuable comments and suggestions for much improved paper and package **clues**! The authors are also grateful to Dr. Vincent Carey at Channing Laboratory, Harvard Medical School, for valuable comments and suggestions. Lazarus and Qiu's research was supported by grant R01 HG003646 from the National Institutes of Health.

References

Asuncion A, Newman DJ (2007). "UCI Machine Learning Repository." URL http://www.ics.uci.edu/~mlearn/MLRepository.html.

Calinski RB, Harabasz J (1974). "A Dendrite Method for Cluster Analysis." *Communications in Statistics-Simulation and Computation*, **3**(1), 1–27.

Chang F, Qiu W, Zamar RH, Lazarus R, Wang X (2010). "**clues**: An R Package for Nonparametric Clustering Based on Local Shrinking." *Journal of Statistical Software*, **33**(4), 1–16. URL http://www.jstatsoft.org/v33/i04/.

Chipman H, Tibshirani R (2006). "Hybrid Hierarchical Clustering with Applications to Microarray Data." *Biostatistics*, **7**(2), 286–301.

Chipman H, Tibshirani R, Hastie T (2008). "**hybridHclust**: Hybrid Hierarchical Clustering." R package version 1.0-3, URL http://CRAN.R-project.org/package=hybridHclust.

Dimtriadou E (2009). "**cclust**: Convex Clustering Methods and Clustering Indexes." R package version 0.6-16, URL http://CRAN.R-project.org/package=cclust.

Fisher RA (1936). "The Use of Multiple Measurements in Taxonomic Problems." *Annals of Eugenics*, **7**, 179–188.

Fraley C, Raftery AE (2002). "Model-based Clustering, Discriminant Dnalysis, and Density Estimation." *Journal of the American Statistical Association*, **97**, 611–631.

Fraley C, Raftery AE (2008). "**mclust**: Model-Based Clustering / Normal Mixture Modeling." R package version 3.4.1, URL http://CRAN.R-project.org/package=mclust.

Hartigan JA, Wong MA (1979). "A K-Means Clustering Algorithm." *Applied Statistics*, **28**, 100–108.

Hennig C (2007). "**fpc**: Fixed Point Clusters, Clusterwise Regression and Discriminant Plots." R package version 1.2-7, URL http://CRAN.R-project.org/package=fpc.

Hornik K (2005). "A CLUE for CLUster Ensembles." *Journal of Statistical Software*, **14**(12), 1–25. URL http://www.jstatsoft.org/v14/i12/.

Hornik K (2009). "**clue**: Cluster Ensembles." R package version 0.3-33, URL http://CRAN.R-project.org/package=clue.

Jain AK, Murty MN, Flynn PJ (1999). "Data Clustering: A Review." *ACM Computing Surveys*, **31**(3), 264–323.

Kaufman L, Rousseeuw PJ (1990). *Finding Groups in Data: An Introduction to Cluster Analysis*. John Wiley & Sons, New York.

Mack YP, Rosenblatt M (1979). "Multivariate K-Nearest Neighbor Density Estimates." *Journal of Multivariate Analysis*, **9**(1), 1–15.

MacQueen JB (1967). "Some Methods for Classification and Analysis of Multivariate Observations." In LM Le Cam, J Neyman (eds.), *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics*, pp. 281–297. University of California Press, Berkeley.

Maechler M, Rousseeuw P, Struyf A, Hubert M (2009). "**cluster**: Cluster Analysis Basics and Extensions." R package version 1.12.1, URL http://CRAN.R-project.org/package=cluster.

Milligan GW, Cooper MC (1986). "A Study of the Comparability of External Criteria for Hierarchical Cluster Analysis." *Multivariate Behavioral Research*, **21**(4), 441–458.

Picard F (2010). "**segclust**: A Package for Segmentation and Segmentation/Clustering." R package version 0.75, URL http://CRAN.R-project.org/package=segclust.

Picard F, Robin S, Lebarbier E, Daudin JJ (2007). "A Segmentation/Clustering Model for the Analysis of Array CGH Data." *Biometrics*, **63**(3), 758–766.

R Development Core Team (2009). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, URL http://www.R-project.org/.

Templ M (2008). "**clustTool**: GUI for Clustering Data with Spatial Information." R package version 1.6.4, URL http://CRAN.R-project.org/package=clustTool.

Wang XG, Qiu WL, Zamar RH (2007). "CLUES: A Non-Parametric Clustering Method Based on Local Shrinking." *Computational Statistics & Data Analysis*, **52**(1), 286–298.

Zhang NR, Siegumund D (2007). "A Modified Bayes Information Criterion with Applications to the Analysis of Comparative Genomic Hybridization Data." *Biometrics*, **63**(1), 22–32.

Affiliation:

Fang Chang, Xiaogang Wang
Department of Mathematics and Statistics
York University
4700 Keele Street, Toronto, Canada
E-mail: changf@mathstat.yorku.ca, stevenw@mathstat.yorku.ca

Weiliang Qiu, Ross Lazarus
Channing Laboratory, Department of Medicine
Brigham and Women's Hospital, and Harvard Medical School
181 Longwood Avenue, Boston, United States of America
E-mail: stwxq@channing.harvard.edu, ross.lazarus@channing.harvard.edu

Ruben H. Zamar
Department of Statistics
University of British Columbia
333-6356 Agricultural Road, Canada
E-mail: ruben@stat.ubc.ca