

A Quantitative Comparison of functional MRI Cluster Analysis¹

Evgenia Dimitriadou^a Markus Barth^{b,2}

Christian Windischberger^c Kurt Hornik^a Ewald Moser^b

^a*Institut für Statistik und Wahrscheinlichkeitstheorie, Technische Universität*

Wien, Austria

^b*Univ.Klinik für Radiodiagnostik, Universität und AKH Wien, Austria*

^c*AG NMR, Institut für Medizinische Physik, Universität Wien, Austria*

Email addresses: `dimi@ci.tuwien.ac.at` (Evgenia Dimitriadou),
`markus.barth@univie.ac.at` (Markus Barth).

¹ This study is funded by the Austrian National Bank Fund (ÖNB-Project No.9201).

² Corresponding Author: Markus Barth, Waehringer Guertel 18-20, A-1090 Vienna, Austria, phone: ++43 -1- 40 400 ext. 1772, fax: ext. 7631, e-mail: `markus.barth@univie.ac.at`

Abstract

The aim of this work is to compare the efficiency and power of several cluster analysis techniques on fully artificial (mathematical) and synthesized (hybrid) fMRI data sets. The clustering algorithms used are hierarchical, crisp (neural gas, self-organizing maps, hard competitive learning, k -means, maximin-distance, CLARA) and fuzzy (c -means, fuzzy competitive learning). To compare these methods we use two performance measures, namely the correlation coefficient and the weighted Jaccard coefficient. Both performance coefficients clearly show that the neural gas and the k -means algorithm perform significantly better than all the other methods using our setup. For the hierarchical methods the ward linkage algorithm performs best under our simulation design. In conclusion, the neural gas method seems to be the best choice for fMRI cluster analysis, given its correct classification of activated pixels (TP) whilst minimizing the misclassification of inactivated pixels (FP scores), and in the stability of the results achieved.

Key words: Clustering Algorithms, fMRI analysis, validation, comparative study, performance coefficients

1 Introduction

Functional Magnetic Resonance Imaging (fMRI) based on Blood Oxygenation Level Dependent (BOLD) signal changes allows assessment of brain activity via local hemodynamic variations over time [37]. fMRI provides one of the optimum combined spatial and temporal resolution methods presently available for non-invasive functional brain mapping. In a typical fMRI experiment external stimuli are presented at intervals of several seconds, causing a change in voxel-signal intensity, delayed and blurred by the hemodynamic response. These data sets can be analyzed in a pixel-by-pixel fashion using model-based statistical methods with estimated Hemodynamic Response Functions (HRF). However, neither are the basic mechanisms such as the coupling between neuronal activation and hemodynamic response exactly known, nor can a number of artifacts be predicted or controlled.

In this context Exploratory Data Analysis (EDA), as a data-driven analysis, is useful as an unbiased analysis method for fMRI data. By using EDA methods it is easier to identify artifacts (such as movement, drifts, or spikes) which can potentially degrade data quality and frustrate further analysis [47]. EDA is also used to estimate brain responses which are either unexpected [48] or where common biophysical models, needed to further classify functional activation, are too complex [5]. This is possible as EDA methods do not need *a priori* information, i.e. exact knowledge of stimulation paradigm, individual

hemodynamic response, or noise distribution; they rather search for groups of time courses which are “different” and, therefore, potentially interesting.

Cluster analysis is an important exploratory data analysis tool that has been used to identify regions with similar patterns of activation [31, 41, 34, 10, 11, 9, 17, 35, 8, 7, 3, 4]. The general motivation for clustering fMRI data is that in general not a single pixel will be activated by a task, rather groups of tens to hundreds of pixels. Although some studies compared single exploratory and non-exploratory methods [9, 10, 11, 22, 8, 28], no comprehensive performance comparison among cluster algorithms has been performed to date.

For our simulation study we chose those algorithms described in the literature which are broadly used and representative of the main clustering algorithmic categories, and, in addition, which are often used for the fMRI analysis (see for example, [17, 21, 36, 33, 22, 13, 7]). Data analysis was performed using these methods on several data sets similar to those encountered in many studies, with different spatial patterns and data distributions, as well as noise characteristics. The stability and reliability of the clustering results was established by performing repeated (50) runs for each simulation.

In order to estimate and validate the performance of all methods we used two performance coefficients (PCs). These PCs were chosen to match the requirements of fMRI. In fMRI the correct detection and classification of the activation cluster is crucial, meaning that in addition to identify all truly ac-

tivated pixels it is also very important to achieve the minimum possible rate of false positives (misclassification of non-activated pixels) so that fMRI activation cluster(s) can be identified by the researcher. The first coefficient used, the correlation coefficient, is a common and intuitive measure which reveals the quality of the cluster by calculating the correlation of the activation cluster center with the reference time course (the mean vector of the truly activated data time courses). A more rigid measure is the Jaccard coefficient (JC), which takes into account absolute scores of the correctly classified activated pixels (true positives; TP), false positives (FP), and false negatives (FN) found in an activation cluster, and thus represents a quantitative measure of the quality of the cluster. To account for the rather small number of TPs compared to the total number of pixels in fMRI, we weighted the scores according to the frequency of a score, which leads to the weighted Jaccard coefficient (wJC).

This paper is organized as follows. In section 2 a description of the generation of the data sets is given, as well as an analytical description of the clustering methods used. Section 2.3 explains the simulation design of the experiments, the preprocessing of the data, and the parameter initialization. Section 3 describes the results and, finally, in section 4 results are discussed and conclusions are drawn.

2 Materials and Methods

2.1 fMRI Data Sets

Quantitative performance assessment was done using two data sets, one simulated and one hybrid. The simulated set was a fully artificially constructed mathematical fMRI phantom with activation introduced [45]. It consisted of a time series (35 instances) of a transversal brain slice (128×128 pixels) with a time invariant texture of 4809 pixels (gray/white matter, ventricles) and three regions of activation (49 pixels, see Figure 1(a)), where the signal intensity increases during “activation” in a box-car-like pattern (5 instances off, 5 on, etc). The relative signal increase upon “activation” in all activated pixels was 6%, 5%, and 4% with 3% baseline Gaussian noise of zero mean value added, resulting in three simulated fMRI data sets with functional CNR (contrast-to-noise ratio) of 2, 1.66, and 1.33 respectively, common values in fMRI of the human brain.

The hybrid (synthesized) data sets were constructed using a baseline in vivo MRI data set with activation added artificially. It consisted of a time series of 140 images with a matrix size of 64×64 pixels. Data acquisition was performed on a 3 Tesla whole-body MR scanner (MedSpec S300, Bruker Biospin, Ettlingen, Germany) using single-shot gradient echo-planar imaging (T_R of 1000 s, effective T_E of 35 ms, slice thickness 4 mm, field-of-view of 23×23 cm²).

The slice chosen was overlaid with 25 pixels of activation (a square of 5×5 , see Figure 1(b)), where the signal intensity increases during “activation” in a box-car-like pattern (20 instances off, 20 on, etc) with a contrast-to-noise ratio of 2, 1.66, and 1.33, where the noise was calculated inside a region within the brain. As the baseline data were sampled *in vivo* the noise characteristics is different from the simulated data set, because *in vivo* noise is known to contain correlated, non-white noise components.

Note that abbreviated names used for the data sets in the tables and figures are *t6*, *t5*, and *t4* for the 3 different CNR levels of the artificial data sets, and *h6*, *h5*, and *h4* for the hybrid data respectively. All data sets used for the simulations are available at

http://www.ci.tuwien.ac.at/research/oenb/oenb_data.html.

2.2 Algorithms

In the following analytical description of the algorithms the names of the methods are written in boldface, and the abbreviations used in the text, and in all tables and figures appear in italics ³.

All our experiments were performed in R [24], a system for statistical com-

³ Note, that variants of some of the methods used in our simulation study exist that are modified specifically to ensure very fast computing times (i.e. very large data sets) or to perform better under several circumstances (i.e. noise characteristics of the data). However, it was our concern to perform a fair comparison of the original algorithms. Once a method has been chosen, it is up to the researcher to use any modifications according to the of requirements her/his study

putation and graphics, which conform to well-known and award-winning S-language for statistics (“GNU S”). R runs under a variety of Unix platforms (such as Linux) and under Windows95/98/ME/2000/NT. It is available freely via CRAN, the Comprehensive R Archive Network, whose master site is at <http://www.R-project.org>. All the clustering algorithms used for the simulation design can be found in R in the base, or in the `cclust` [12], `e1071` [14], `cluster` [40], and `GeneSOM` [49] packages.

2.2.1 Hierarchical Clustering

These methods perform an hierarchical cluster analysis [1] using a set of dissimilarities for the N objects being clustered. Initially, each object is assigned to its own cluster and the algorithm proceeds iteratively, joining the two most similar clusters at each stage and continuing until a single cluster remains. At each stage distances between clusters are computed by the Lance-Williams dissimilarity update formula according to the particular clustering method being used. Given a set of N items to be clustered, and an $N \times N$ distance (or dissimilarity) matrix, the basic process of Johnson’s (1967) hierarchical clustering is:

- (1) Start by assigning each data item (data point) to its own cluster, so that if N items are being considered there are N clusters, each containing just one item. Let the distances (similarities) between the clusters equal the distances (dissimilarities) between the items they contain.

- (2) Find the closest (most similar) pair of clusters and merge them into a single cluster, reducing the number of clusters by one.
- (3) Compute distances between the new cluster and each of the old clusters.
- (4) Repeat steps 2 and 3 until all items are clustered into a single cluster of size N .

Step 3 can be achieved in different ways, distinguishing single-linkage from complete-linkage and other linkage methods. In **single-linkage** clustering (also called the connectedness or minimum method), we consider the distance between clusters to be equal to the shortest distance from any member of one cluster to any member of the other cluster. In **complete-linkage** (*complete*) clustering (also called the diameter or maximum method), we consider the distance between clusters to be equal to the greatest distance from any member of one cluster to any member of the other cluster. The complete linkage method finds similar clusters. **Ward's minimum variance** method (*ward*) aims at finding compact, spherical clusters. Here, those two clusters are merged which give the minimum increase in total within group error sum of squares, calculated from the distance between each data point in a cluster k and the cluster center.

2.2.2 Crisp Clustering

Maximin-Distance (*maximin*): The *maximin*-distance algorithm [44] is a simple heuristic procedure based on the euclidean distance concept. It is some-

times used as a cluster center initialization for other algorithms such as c -means (e.g. used by Evident(TM) [25, 42, 39]) to assure stability of the result and to avoid random initialization.

The procedure to create the clusters is:

- (1) Initialize the set C to contain k ($k \ll N$) units c_j : $C = \{c_1, c_2, \dots, c_k\}$ with only one center vector $w_{c_j} \in \mathbb{R}^d$ chosen randomly from the data set and compute the distance between this point and all others in the set.
- (2) Determine the farthest point to be the next cluster center.
- (3) Compute the distance from each remaining data point to the centers already found. For every pair of these computations, save the minimum distance and select the maximum of these minimum distances. If this distance is greater than a predefined threshold (for example, at least one half of the distance between the 2 centers initially selected) then that point becomes a new center.
- (4) Repeat until the new maximum distance at a particular step fails to satisfy the condition for the creation of the new cluster, or until the desired number of clusters is reached.

k -Means (*kmeans*): The k -means algorithm [29, 19] is one of the classic statistical clustering methods. As opposed to the competitive learning variants, this is an off-line method, i.e., the center updates are based on the entire training sample:

- (1) Initialize the set C to contain k ($k \ll N$) units (centers) c_j : $C = \{c_1, c_2, \dots, c_k\}$ with center vectors $w_{c_j} \in \mathbb{R}^d$ chosen randomly from the data set and assign the data to the clusters with the corresponding initialized centers.
- (2) Compute the centers of all current clusters.
- (3) Generate a new partition of the data set by assigning each data item to the closest cluster center.
- (4) If the partition changes compared to the last iteration, go to step 2, else stop.

CLARA (*clara*): The clustering of a set of data points is carried out in two steps. First, a sample is drawn from the data and is clustered into k clusters using the k -medoid method (PAM) [26], which gives k representative points (medoids) of the data set. Then each data point which does not belong to the sample is assigned to the nearest representative. This yields a partition of the whole data set. A quality measure of this partition is obtained by calculating the average distance between each data point of the data set and its center (medoid). After all the samples have been drawn and clustered, the one with the lowest average distance is obtained. The clustering for the samples takes place in two phases (PAM algorithm). At first, an initial clustering is obtained by the successive selection of representative points until a number of k ones has been found:

- (1) Take a point x_i (not yet selected).

- (2) Consider a non-selected x_j point and calculate the difference between its dissimilarity D_j with the most similar previously selected one, and its dissimilarity $d(i, j)$ with the point x_i .
- (3) If the difference is positive, then x_j contributes to the decision to select x_i . $C_{ij} = \max(D_j - d(j, i), 0)$
- (4) The total gain by selecting x_i is: $\sum_j C_{ji}$
- (5) Choose x_i (not yet selected) which maximizes $\max_i \sum_j C_{ji}$

In the second phase, the set of representative points is improved by considering all pairs of points (x_i, x_l) for which point x_i has been selected and point x_l has not. Thus, this is the effect on the value of clustering when a swap takes place (for more details see [26]). If we consider the euclidean distance as the dissimilarity measure, this step provides the main difference between the known k -means algorithm and PAM. An advantage of the algorithm is that it can be used for large data sets using sampling, reducing data storage requirements and the number of calculations. The storage requirement of *clara* computation (for small k) is about $O(n * p) + O(j^2)$ where $j =$ sample size, and $(n, p) = \dim(x)$. The CPU computing time (again neglecting small k) is about $O(n * p * j^2 * N)$, where N is the number of samples.

Hard Competitive Learning (*hardcl*): Hard competitive learning is a well-known online stochastic gradient descent algorithm for minimization of the average distance of a given set of data to its closest center. See, e.g., reference [19] for a survey on competitive learning methods. *hardcl* is the simplest on-

line clustering algorithm, where only one output unit (the cluster center) is the winner (the closest to the data point) for each given data point, and the vector of the winner moves toward the vector of the given point. For *hardcl*, the set of cluster centers $C = \{c_1, c_2, \dots, c_k\}$ are seen as output units of a neural network, which adapt every time a new data point is presented.

- (1) Initialize the set C to contain k ($k \ll N$) units c_j : $C = \{c_1, c_2, \dots, c_k\}$, with center vectors $w_{c_j} \in \mathbb{R}^d$ chosen randomly from the data set. Set the iteration counter t to zero.
- (2) Draw a pattern x_i from the data set.
- (3) Determine the winner $s(x_i)$: $s(x_i) = \arg \min_{c \in C} \|x_i - w_c\|$
- (4) Move the center vector of the winner along the gradient of $\|x_i - w_{s(x_i)}\|$ toward x_i . In the case of the Euclidean norm this is $\Delta w_{s(x_i)} = \varepsilon_t(x_i - w_{s(x_i)})$, where ε_t is a suitable chosen learning rate.
- (5) Set $t := t + 1$; if $t < t_{\max}$, return to step 2.

Neural Gas (*ngas*): The neural gas algorithm [30] is an example of a soft competitive learning algorithm, i.e., not only the winner unit is adapted after the presentation of an input data item, but also some, or all, of the remaining units. In contrast to the *som* algorithm, no topology of a fixed dimensionality is imposed on the network. The neural gas algorithm sorts for every input item (data point) the units of the network to a rank order according to the distance of the center vectors to the input point. After sorting all units are adapted according to their rank order.

We present pseudo-code of the algorithm:

- (1) Initialize the set C to contain N units c_j

$$C = \{c_1, c_2, \dots, c_k\}$$

with center vectors $w_{c_j} \in \mathbb{R}^d$ chosen randomly from the data set.

Set the iteration counter t to zero.

- (2) Draw an item x_i from the data set.
- (3) Order all elements of C according to their distance to x_i , i.e., find the sequence of indices $(l_0, l_1, \dots, l_{k-1})$ so that the center vector w_{l_0} is closest to the pattern, w_{l_1} the second closest and etc. We denote the number associated with w_l as $m_l(x_i, C)$.
- (4) Adapt the center vectors according to

$$\Delta w_l = \varepsilon_t h_\lambda(m_l(x_i, C)) \nabla \|(x_i - w_l)\|$$

where

$$\begin{aligned} \lambda_t &= \lambda_{\text{ini}} (\lambda_{\text{fin}} / \lambda_{\text{ini}})^{(t/t_{\text{max}})} \\ \varepsilon_t &= \varepsilon_{\text{ini}} (\varepsilon_{\text{fin}} / \varepsilon_{\text{ini}})^{(t/t_{\text{max}})} \\ h_\lambda(m) &= \exp(-m/\lambda_t). \end{aligned}$$

and $\nabla \|(x_i - w_l)\|$ denotes the gradient of the distance function, cf. the description of the *hardcl* algorithm.

- (5) Set $t := t + 1$; if $t < t_{\text{max}}$ continue with step 2.

Suitable initial values $(\lambda_{\text{ini}}, \varepsilon_{\text{ini}})$ and final values $(\lambda_{\text{fin}}, \varepsilon_{\text{fin}})$ have to be chosen

for the iteration-dependent parameters λ_t and ε_t .

Self Organizing Maps (*som*): The SOM algorithm [27] is a soft-competitive learning algorithm where an additional neighborhood structure is imposed on the cluster centers. For this method we have a network of a fixed dimensionality d which has to be chosen in advance. One advantage of a fixed network dimensionality is that such a network defines a mapping from the n -dimensional input space to the d dimensional structure. This makes it possible to get a low-dimensional representation of the data, which may be used for visualization purposes. We give a short pseudo-code description of the algorithm:

- (1) Initialize the set C to contain k ($k \ll N$) units c_j .
- (2) Define a neighborhood relation on the cluster centers. Usually this neighborhood relation is a rectangular grid. Set the iteration counter t to zero.
- (3) Draw an item x_i from the data set.
- (4) Determine the winner $s(x_i)$:

$$s(x_i) = \arg \min_{c \in C} \|x_i - w_c\|$$

- (5) Adapt the center vectors according to

$$\Delta w_l = \varepsilon_t \phi(l, s(x_i))(x_i - w_l)$$

where ε_t is the learning rate and $\phi(l, s(x_i))$ is a neighborhood function.

Due to this neighborhood function, not only the winning center vector

learns, but also center vectors in its topological neighborhood, whose size decreases during the learning process.

- (6) Set $t := t + 1$; if $t < t_{\max}$ continue with step 3.

2.2.3 Fuzzy Clustering

C-Means (*cmeans*): The *c*-means algorithm (or fuzzy *k*-means [38]) is one of the classic fuzzy clustering methods. As opposed to its competitive learning variants, this is an off-line method, i.e., the center updates are based on the whole training sample:

- (1) Fix the number of clusters k , $2 \leq k < N$ where N is the number of data points. Fix q , $1 < q < \infty$, where q is the defuzzification parameter.
- (2) Set the iteration counter t to zero and initialize the cluster centers v_i .
- (3) Calculate the fuzzy k -partition U from

$$U_{t+1} = u_{ij, t+1} = \left[\sum_{l=1}^k \left(\frac{\|x_i - v_{j,t}\|}{\|x_i - v_{l,t}\|} \right)^{\frac{2}{q-1}} \right]^{-1}$$

and update the cluster centers according to

$$V_{t+1} = v_{t+1} = \frac{\sum_{i=1}^N (u_{ij, t+1})^q x_i}{\sum_{i=1}^N (u_{ij, t+1})^q}$$

where $1 \leq i \leq N$ and $1 \leq j \leq k$.

- (4) If $\|U_t - U_{t-1}\| \leq \epsilon$ then stop, else $t = t + 1$.

Unsupervised Fuzzy Competitive Learning (*ufcl*): This is the on-line version of the fuzzy *c*-means algorithm [38]. Every time a new point is chosen

from the data set the membership value for this point is calculated and the centers are updated according to

$$v_{j,t} = v_{j,t-1} + \alpha_t u_{ij,t}^q [x_i - v_{j,t-1}]$$

where $\alpha_t = \alpha_0(1 - t/T)$.

2.3 Simulation Design

Preprocessing as well as parameter initializations were chosen identically for all data sets used. At the preprocessing stage the design matrix consisted of a 2-dimensional array of the time series of every pixel (object), where the time is discrete and handled as the feature space of the pixels. Preprocessing included thresholding of the data to segment the brain. For the 3 simulated data sets ($t4$, $t5$, $t6$) this resulted in 4809 pixels (49 true positives) over 35 time points, and in 1212 (25 true positives) pixels over 140 time points for the 3 hybrid data sets ($h4$, $h5$, $h6$). To remove the influence of different intensity levels, which are not relevant for detection of activation, the median for every pixel over the time was subtracted. For the *som*, the data had to be normalized to mean 0 and variance 1.

For all clustering methods several parameter values are needed during initialization. The hyperparameter tuning of the i.e. defuzzification parameter (for the fuzzy methods of value 1.1), or learning rate (for the on-line methods)

is based on previous fMRI experience and other benchmarking experiments. The euclidean distance was used as the dissimilarity measure between the pixels. The learning rate used, was not constant, but rather time-dependent (iteration-dependent). The number of clusters has to be defined a priori for all the methods and we chose 5, 10, and 20. Hierarchical methods do not need any cluster initialization; here the number of clusters was defined at the end of the run, when the appropriate cut was made according to the number of clusters desired.

For all clustering methods except for *clara* each simulation run consisted of 50 repetitions of the algorithm. For *clara* 20 repetitions were made, using 5 to 30 samples. The “bubble” neighborhood function type was chosen for the *som* method. For a direct comparison with the other methods (concerning the initial number of clusters (INC)), the initialized grid contained 2×2 units, 3×3 units, and 5×4 units, resulting in 4, 9 and 20 centers, respectively (compared to the 5, 10, and 20 used for the other methods). For the initialization of the map a random data sample was used. Note that for all methods data were randomly mixed before entering the clustering procedures. We define the activation cluster as the one for which the maximum absolute number of true positive pixels was found by the algorithm. For the fuzzy clustering methods a defuzzification had to be performed first, assigning a crisp membership of a pixel to a cluster (a value of 1) to pixels with membership greater or equal than 0.5, and of 0 for those remaining (i.e. membership smaller than 0.5).

2.4 Performance Coefficients

For all repeated runs of the algorithms two association coefficients were used to validate and compare the performance of the methods. A natural and intuitively appealing approach in choosing those coefficients is to use a measure revealing the time course similarity between the cluster of interest of the clustering result (partition result) and the reference class (the class including only all truly activated pixels).

2.4.1 Correlation Coefficient (CC)

We calculate the correlation between the center of the activation cluster identified and the center of the reference activation class (the arithmetic mean of all the activated pixels):

$$CC = \frac{\sum(c_i - \bar{c})(c_i^* - \bar{c}^*)}{\sqrt{\sum(c_i - \bar{c})^2 \sum(c_i^* - \bar{c}^*)^2}},$$

where c is the center (vector) of the activation cluster identified and c^* the reference center.

The correlation coefficient should exhibit the quality of the cluster represented by its center towards that of the reference.

Another approach to assessing the similarity between 2 cluster results is simply to count the total number of relevant matches between the partitions. We

describe the case where a clustering partition is presented by a binary vector, where the class/cluster labeled 1 represents the activation class/cluster, and 0 all remaining ones. Two factors are important: first, it would be logical not to allow TN (true negatives) to contribute to the association measure, since the true negatives (correct classifications of the non-activated pixels) do not necessarily explain and characterize the quality of an activation cluster (they can be activated pixels not in the activation but in any remaining cluster, and thus assigned with the 0 label of no activation); second, to weight matches and mismatches according to the importance and the occurrence of the class in the data. The desire that rare classes receive extra weight appears frequently in the biological literature. To achieve this it seems logical to assign each class a weight which is a function of the frequency with which the class occurs in the data set. If n is the number of data (pixels) in the data set and n_i the number of data (pixels) in the i th class, then the probability that a randomly chosen data falls in the class i is n_i/n . The probability for two randomly chosen data would be $P_{ii} = (n_i/n)^2$ and the probability⁴ that one falls in i and the other in j is $P_{ij} = (2n_in_j)/n^2$. Thus, the probability of any pair of classes occurring can be calculated. Because this probability of rare classes is usually low in fMRI, any inverse function of it can be used as a suitable weight.

⁴ The factor of 2 reflects the fact that the first data unit may fall in either of the two classes so that there are two ways for the event to occur.

2.4.2 Weighted Jaccard Coefficient (*wJC*)

We used the weighted Jaccard coefficient (*wJC*) [1], taking the Jaccard coefficient [2] ($JC = a/(a + b + c)$) and adding to a , b , and c weights which are the inverse functions of the probabilities of a TP, FN, and FP respectively, under the hypothesis that each pixel has the same probability arising in a cluster or of belonging to the activation/non-activation class.

$$wJC = \frac{a + \frac{1}{\mathbb{P}(a)}}{\left(a + \frac{1}{\mathbb{P}(a)} + b + \frac{1}{\mathbb{P}(b)} + c + \frac{1}{\mathbb{P}(c)}\right)},$$

where a is the number of TPs, b the number of FNs, and c the number of FPs, and their probabilities $\mathbb{P}(a) = (a + c) * (a + b)/n^2$, $\mathbb{P}(b) = (b + d) * (a + b)/n^2$, and $\mathbb{P}(c) = (a + c) * (c + d)/n^2$, respectively, for d TN pixels and n pixels in the data set.

The weighted Jaccard coefficient should thus provide a quantitative measure of the quality of the activation cluster because it emphasizes the scores of the TPs.

3 Results

The results of the simulation study and the performance of the algorithms are presented by:

- (1) ranking the methods according to their performance evaluated by the 2

coefficients (see Tables 1, 2, 3, 4, 5).

- (2) displaying box-and-whisker plots for both performance coefficients (wJC and CC) averaged over all runs and CNRs (see Figures 2, and 3).
- (3) displaying box-and-whisker plots for the TP and FP in the activation cluster for all the repeated runs (see Figures 4, 5, 6, 7, 8, and 9)

Tables 1, 2, 3, and 4 show the value of the performance coefficients and the corresponding rank (in parentheses) within each column for the non-hierarchical methods, i.e. the best overall performance over data sets (varying noise characteristics and CNR) and number of INCs, and Table 5 for the non-hierarchical methods averaged over the different INCs, as they cannot be initialized in this way. Each coefficient value in every table cell corresponds to the median of the 50 coefficient values (for the 50 repetitions of the methods). Within each table the methods are ranked in descending order according to the mean PC values over each row.

For the non-hierarchical methods the tables with the CC values clearly show that two clustering methods, namely *ngas* and *kmeans*, have high CC values and stable performance for both the different levels of noise characteristics and CNR values. These two methods perform excellently as long as the INCs is sufficiently large (i.e. 20 clusters). *hardcl* performs similarly well for the artificial data sets. Looking at wJC ranks for the non-hierarchical methods, again *ngas* and *kmeans* are within the top performing methods, underperforming *maximin* for the artificial data sets only; however, *maximin* performs worst

on the hybrid data. As expected PC values decrease with decreasing CNR level. It can also be seen that, in general, PC values increase with increasing INCs. However, the best performing methods show very high PCs even on data sets with a very low CNR of 1.33. The performance of the methods over the two levels of noise characteristics is quite heterogeneous. In general, PCs are higher for the artificial data set - especially for *maximin* and *hardcl* - but some methods perform comparably (*som*, *clara*) and some even better for the hybrid data sets in both PC values and ranks (*ufcl*, *cmeans*).

The overall best performing method for the hierarchical methods is ward linkage. Complete linkage performs similarly for the artificial data set, but not for the hybrid data, and single linkage fails almost completely. The best hierarchical method (*ward*) shows a similar or even better performance compared to best performing non-hierarchical method; however, one should keep in mind that hierarchical methods are very demanding on memory and the computation of the dissimilarity matrix can be expensive and unfeasible for large data sets.

The box-and-whisker plots in Figures 2, and 3 give an overview of the variance over the runs and, thus, of the stability of the various algorithms. Box-and-whisker plots have been used due to their intuitive and powerful visual representation of typical characteristics of the data⁵. They highlight few, but

⁵ In a box-and-whisker plot, the median is shown (as full dot), the box ends at the first Q_1 and third Q_3 quartiles, the whiskers are extended until 3/2 times the interquartile ranges of Q_1 and Q_3 , and outliers are shown as open dots.

important features of the data, which makes it easy to present data - independent of the number of data values - in an organized way. From these plots it is clear that an INC value of 5 gives relatively small variance but accompanied by a low median PC, whereas an INC of 10 gives high variance in general, especially for the hybrid data. An INC of 20 gives very low variance for the best performing methods (*ngas*, *kmeans*), followed by *som* which show relatively good stability and high PC values; other methods (*clara*, *maximin*, *hardcl*, *ufcl*) prove to be unstable in general.

Figures 4, 5, 6, 7, 8, and 9 show a very detailed view of the results. These plots show the median and the dispersion of the classification results (TPs and FPs) over the 50 runs for each data set. *maximin*-initialized results (used by Evident (TM) [25, 42, 39]) are indicated with an asterisk (“*”) for comparison reasons with the random initialized results. In many cases, this kind of initialization performs superiorly to the median of the randomized approach. These plots also show that *maximin* achieves high *wJC* values for the artificial data only, due to the very low number of FPs, while the FP variability over the 50 runs is higher for the other methods. Also here *ngas* has the lowest dispersion of the TP and FP scores over the repetitions of the methods, whilst conserving a high number of TPs and a low number of FPs.

4 Discussion

To our knowledge this is the first study which extensively examines the performance of most common clustering algorithms on fMRI data sets based on association measures (PCs) using a broad simulation design. We have varied several important design factors such as noise characteristics, CNR level, and initial number of clusters to compare and validate the simulation results. 50 runs (repetitions) were performed for all 11 clustering methods and for all combinations of design factors. This enables the stability and average performance to be estimated. Single runs would be biased by the random initialization.

A good performance of a cluster analysis method in fMRI requires correct classification of the activation cluster. For a critical validation and mutual comparison of the performance of clustering methods it is necessary to use measures which reflect the ability of the algorithm to detect the “ground truth”. After the clusters are identified by the algorithm, the researcher has to select the activation cluster(s) and other rather “interesting” clusters (those containing artifacts or other useful information) based on knowledge of the stimulation paradigm. An important and necessary requirement for EDA methods is, therefore, that fMRI-typical activation cluster(s) are represented relatively “cleanly” and “unspoiled”. The interpretation of clustering results is dependent on the numbers of TPs and FPs in a cluster, which have an important impact as FPs “spoil” the average time course of a cluster (cluster center)

used to classify the cluster e.g., as activated. To obtain a reasonable performance comparison of the different algorithms we have to find appropriate performance measures which consider these requirements. We therefore use a qualitative and a quantitative association measure to assess the performance of the methods and validate the quality of the clustering partitions. General considerations and conclusions on the methods applied on several data sets can be drawn, but a decision as to the best performing algorithm might be a question of the preferred condition/criterion to be satisfied.

Although clustering proves to be a flexible tool for fMRI analysis there are several drawbacks to these methods. Firstly, most clustering approaches make assumptions about cluster shape and size so that the algorithms might artificially induce a topology which deviates from the real structure of the data. This might lead to misinterpretations of the analysis results [32, 23, 46, 15, 16]. The optimization techniques applied by the algorithms [33, 50], the metric [20, 22, 43], as well as the random initializations of the methods [18, 23, 32, 6], may constitute other drawbacks, resulting in local minima and instable results. The random mixing of the data points, as well as the random initialization of the cluster centers prevent the algorithm from reaching the global minimum of its error function, often resulting in local minima. We therefore performed simulations using many (50) runs and varying parameters such as the initial number of clusters and data set features such as noise characteristics and CNR ratio.

Our results suggest that it is important to use the most suitable algorithms, as otherwise the quality of clustering results might be poor. In our study, it emerges that *ngas* is an appropriate and flexible method for the fMRI data segmentation. *ngas* performs very stably over runs and relatively independently of data set characteristics. In general, the *ngas* methodology - due to its on-line, soft competitive centers update - decreases the dependency on random initializations, being able to emerge from poor local minimum, where it was probably initialized. It is also obvious that the performance of the fuzzy algorithms is deteriorated in the case of the artificial data sets. This is due to the fact that they have a “smoother” noise profile, which seems to favor non-fuzzy decisions. However, for the hybrid data sets the fuzzy membership assigned to the pixels returns a better classification, especially for the *ufcl* method. Most methods impose a topology of rather compact and spherical clusters which seems to be the appropriate structure for the activation class which may explain their good performance. This is to be expected as we have to deal with a relative small and concentrated number of activated pixels, which can be adequately represented by the compact and spherical clusters generated by most methods. For the hierarchical methods, the joining of the clusters based on the shortest distance between their members (single linkage) proves to be an inadequate criterion for fMRI analysis, as the large number of inactivated pixels bias the clusters. For the same reason, complete linkage returns better results, whilst the ward method, based on the concept of the within-sum-of-squares error (like *kmeans*), performs better on average.

In conclusion, *ngas* seems to be the best choice when looking for the optimal algorithm for unsupervised non-hierarchical fMRI cluster analysis. It performed best in both TP and FP scores and, in addition, it behaved stably. When data sets are reasonably small and the usage of hierarchical algorithms is feasible ward linkage seems to be the most appropriate choice. Since the purpose of this study is to focus on algorithm performance we did not use possibly distorting features such as merging [15], voting [16], or special initialization [42, 25, 39]. Once the most suitable algorithms are used these features can be added to further improve analysis performance.

References

- [1] Michael R. Anderberg. *Cluster Analysis for Applications*, chapter Hierarchical Clustering Methods, page 131. Academic Press, N.Y., 1973.
- [2] Michael R. Anderberg. *Cluster Analysis for Applications*, chapter Measures of Association among Variables, page 89. Academic Press, N.Y., 1973.
- [3] M. Barth, M. Diemling, and E. Moser. Modulation of signal changes in gradient-recalled echo functional MRI with increasing echo time correlate with model calculations. *Magn. Reson. Imag.*, 15:745–752, 1997.
- [4] M. Barth, J.R. Reichenbach, R. Venkatesan, E. Moser, and E.M. Haacke. High resolution, multiple gradient-echo functional MRI at 1.5 T. *Magn. Reson. Imag.*, 17:321–329, 1999.

- [5] M. Barth, C. Windischberger, M. Klarhöfer, and E. Moser. Characterization of BOLD activation in multi-echo fmri data using fuzzy cluster analysis and a comparison with quantitative modeling. *NMR in Biomedicine*, 14:484–489, 2001.
- [6] Markus Barth, Evgenia Dimitriadou, Kurt Hornik, and Ewald Moser. Comparison of clustering methods in fMRI analysis by ranking association coefficients. In *Proc. 11th Scientific Meeting of the Int. Society of Magn. Res. in Medicine*, 2003. accepted.
- [7] Markus Barth, Evgenia Dimitriadou, Christian Windischberger, Kurt Hornik, and Ewald Moser. AveSure - statistical validation of fMRI clustering results. In *NeuroImage*, volume 16, page S371, 2002.
- [8] R. Baumgartner, L. Ryner, W. Richter, R. Summers, M. Jarmasz, and R. Somorjai. Comparison of two exploratory data analysis methods for fMRI: fuzzy clustering vs. principal component analysis. *Magnetic Resonance Imaging*, 18:89–94, 2000.
- [9] R. B. Baumgartner, C. Windischberger, and E. Moser. Quantification in functional Magnetic Resonance Imaging: Fuzzy clustering vs. correlation analysis. *Magnetic Resonance Imaging*, 16(2):115–125, 1998.
- [10] A. Baune, F. T. Sommer, M. Erb, D. Wildgruber, B. Kardatzki, G. Palm, and W. Grodd. Dynamical cluster analysis of cortical fMRI activation. *NeuroImage*, 9:477–489, 1999.
- [11] K. H. Chuang, M. J. Chiu, C. C. Lin, and J. H. Chen. Model-free functional MRI analysis using kohonen clustering neural network and fuzzy

- clustering. *IEEE Trans. Med. Imag.*, 18:1117–1128, 1999.
- [12] Evgenia Dimitriadou. `cclust` – convex clustering methods and clustering indexes. R package, Version 0.6-9, 2002. Available from <http://cran.R-project.org>.
- [13] Evgenia Dimitriadou, Markus Barth, Christian Windischberger, Ewald Moser, and Kurt Hornik. An explorative concept for classifying activation in multi-echo fMRI data. In *Magnetic Resonance, Materials in Physics, Biology and Medicine*, volume 15, page 163, Cannes, France, August 2002. 19th Annual Meeting of the European Society for Magnetic Resonance in Medicine and Biology (ESMRMB 2002).
- [14] Evgenia Dimitriadou, Kurt Hornik, Friedrich Leisch, David Meyer, and Andreas Weingessel. `e1071` – misc functions of the Department of Statistics (e1071), TU Wien. R package, Version 1.3-5, 2002. Available from <http://cran.R-project.org>.
- [15] Evgenia Dimitriadou, Andreas Weingessel, and Kurt Hornik. Voting-Merging: An ensemble method for clustering. In Georg Dorffner, Horst Bischof, and Kurt Hornik, editors, *Proceedings of the International Conference on Artificial Neural Networks (ICANN 01)*, volume LNCS 2130, pages 217–224, Vienna, Austria, August 2001.
- [16] Evgenia Dimitriadou, Andreas Weingessel, and Kurt Hornik. A combination scheme for fuzzy clustering. *International Journal of Pattern Recognition and Artificial Intelligence*, *World Scientific*, 16(7):901–912, 2002.

- [17] Peter Filzmoser, Richard Baumgartner, and Ewald Moser. A hierarchical clustering method for analyzing functional MR images. *Magnetic Resonance Imaging*, 17(6):817–826, 1999.
- [18] Harald Fischer and Jürgen Hennig. Neural network-based analysis of MR time series. *Magnetic Resonance in Medicine*, 41:124–131, 1999.
- [19] Bernd Fritzsche. Some competitive learning methods, April 5 1997.
<http://www.neuroinformatik.ruhr-uni-bochum.de/ini/VDM/research/gsn/>.
- [20] X. Golay, S. Kollias, D. Meier, A. Valavanis, and P. Boesiger. Fuzzy membership vs. probability in cross correlation based fuzzy clustering of fMRI data. In *Proc. of the 3rd Int. Conference on Functional Mapping of the Human Brain, NeuroImage*, volume 3, page S481, 1997.
- [21] Xavier Golay, Spyros Kollias, Gautier Stoll, Dieter Meier, Anton Valavanis, and Peter Boesiger. A new correlation-based fuzzy logic clustering algorithm for fMRI. *Magnetic Resonance in Medicine*, 40:249–260, 1998.
- [22] C. Goutte, P. Toft, E. Rostrup, F. Nielsen, and L. K. Hansen. On clustering fmri time series. *NeuroImage*, 9(3):298–310, 1999.
- [23] Cyril Goutte, Lars Kai Hansen, Matthew G. Liptrot, and Egill Rostrup. Feature-space clustering for fMRI meta-analysis. *Human Brain Mapping*, 13(3):165–183, 2001.
- [24] Ross Ihaka and Robert Gentleman. R: A language for data analysis and graphics. *Journal of Computational and Graphical Statistics*, 5(3):299–314, 1996.
- [25] M. Jarmasz and R. Somorjai. Exploring regions of interest with cluster

- analysis (EROICA) using a spectral peak statistic for selecting and testing the significance of fMRI activation time-series. *Artificial Intelligence in Medicine*, special issue, 2002.
- [26] L. Kaufman and P. J. Rousseeuw. *Finding Groups in Data*, chapter Clustering Large Applications, page 126. Wiley, 1990.
- [27] Teuvo Kohonen. *Self-Organization and Associative Memory*. Springer, Berlin, third edition, 1989.
- [28] Nicholas Lange, Stephen C. Strother, Jon R. Anderson, Finn A. Nielsen, Andrew P. Holmes, Thomas Kolenda, Robert Savoy, and Lars Kai Hansen. Plurality and resemblance in fMRI data analysis. *NeuroImage*, 10:282–303, 1999.
- [29] Yoseph Linde, Andrs Buzo, and Robert M. Gray. An algorithm for vector quantizer design. *IEEE Transactions on Communications*, COM-28(1):84–95, January 1980.
- [30] Thomas M. Martinez, Stanislav G. Berkovich, and Klaus J. Schulten. “Neural-Gas” network for vector quantization and its application to time-series prediction. *IEEE Transactions on Neural Networks*, 4(4):558–569, July 1993.
- [31] M. McIntyre, A. Wennerberg, R. Somorjai, and G. Scarth. Activation and deactivation in functional brain images. In *Proc. of the 2nd Int. Conference on Functional Mapping of the Human Brain*, *NeuroImage*, volume 3, page 582, 1996.
- [32] Ulrich Möller, Marc Ligges, Petra Georgiewa, Carolin Grünling,

- Werner A. Kaiser, Herbert Witte, and Bernhard Blanz. How to avoid spurious cluster validation? a methodological investigation on simulated and fMRI data. *NeuroImage*, 17:431–446, 2002.
- [33] Ulrich Möller, Marc Ligges, Carolin Grünling, Petra Georgiewa, Werner A. Kaiser, Herbert Witte, and Bernhard Blanz. Pitfalls in the clustering of neuroimage data and improvements by global optimization strategies. *NeuroImage*, 14:206–218, 2001.
- [34] E. Moser, M. Diemling, and R. Baumgartner. Fuzzy clustering of gradient-echo functional MRI in the human visual cortex. part ii: Quantification. *J. Magn. Reson. Imag.*, 7(6):1102–1108, 1997.
- [35] Ewald Moser, Richard Baumgartner, Markus Barth, and Christian Windischberger. Explorative signal processing in functional MR imaging. *Int. J. Imag. Syst. Technol.*, 10:166–76, 1999.
- [36] Shing-Chung Ngan and Xiaoping Hu. Analysis of functional magnetic resonance imaging data using self-organizing mapping with spatial connectivity. *Magnetic Resonance in Medicine*, 41:939–946, 1999.
- [37] S. Ogawa, R.S. Menon, D.W. Tank, S.G. Kim, H. Merkle, J.M. Ellermann, and K. Ugurbil. Functional brain mapping by blood oxygenation level-dependent contrast magnetic resonance imaging. A comparison of signal characteristics with a biophysical model. *Biophys. J.*, 64:803–12, 1993.
- [38] Nikhil R. Pal, James C. Bezdek, and Richard J. Hathaway. Sequential competitive learning and the fuzzy c -means clustering algorithm. *Neural Networks*, 9(5):787–796, 1996.

- [39] Nicolino J. Pizzi, Rodrigo A. Vivanco, and Ray L. Somorjai. EvIdentTM: a functional magnetic resonance image analysis system. *Artificial Intelligence in Medicine*, 21:263–269, 2001.
- [40] Peter Rousseeuw, Anja Struyf, and Mia Hubert. `cluster` – functions for clustering (by rousseeuw et al.). R package port by Kurt Hornik and Martin Maechler, Version 1.6-4, 2002. Available from <http://cran.R-project.org>.
- [41] G. Scarth, A. Wennerberg, R. Somorjai, and T. Hindermarsh. The utility of fuzzy clustering in identifying diverse activations in fMRI. In *Proc. of the 2nd Int. Conference on Functional Mapping of the Human Brain, NeuroImage*, volume 2, page S89, 1996.
- [42] R. Somorjai and M. Jarmasz. *Exploratory Analysis and Data Modelling in Functional Neuroimaging*, chapter Chapter 2: Exploratory Analysis of fMRI Data by Fuzzy Clustering - Philosophy, Strategy, Tactics, Implementation. MIT Press, 2002.
- [43] P. Toft, L. K. Hansen, F. Nielsen, S. C. Strother, N. Lange, N. Morch, C. Svarer, O. Paulson, R. Savoy, B. Rosen, E. Rostrup, and P. Born. On clustering of fMRI time series. In *Proc. of the 3rd Int. Conference on Functional Mapping of the Human Brain, NeuroImage*, volume 3, page S456, 1997.
- [44] J. T. Tou and R. C. Gonzalez. *Pattern Recognition Principles*, chapter Pattern Classification by Distances Functions, page 92. Addison-Wesley, 1974.

- [45] W. Backfrieder, R. Baumgartner, and M. Samal et al. Quantification of intensity variations in fMRI using rotated principal components. *Phys. Med. Biol.*, 41:1425–1438, 1996.
- [46] Andreas Weingessel, Evgenia Dimitriadou, and Kurt Hornik. A voting scheme for cluster algorithms. In Gerald Krell, Bernd Michaelis, Detlef Nauck, and Rudolf Kruse, editors, *Neural Networks in Applications, Proceedings of the Fourth International Workshop NN'99*, pages 31–37, Otto-von-Guericke University of Magdeburg, Germany, 1999.
- [47] C. Windischberger, M. Barth, C. Lamm, L. Schroeder, H. Bauer, R.C. Gur, and E. Moser. Fuzzy cluster analysis of high-field functional MRI data. *AIM*, 2002, in press.
- [48] C. Windischberger, C. Lamm, H. Bauer, and E. Moser. Human motor cortex activity during mental rotation. *NeuroImage*, page in press, 2003.
- [49] Jun Yan. GeneSOM – self-organizing map. R package, Version 0.2-5, 2002. Available from <http://cran.R-project.org>.
- [50] K. Zeger, J. Vaisey, and A. Gersho. Globally optimal vector quantizer design by stochastic relaxation. *IEEE Trans. Signal Process.*, 40:310–322, 1992.

Figure Captions

Figure 1. Activation of the Simulation Data Sets

Figure 2. weighted Jaccard Coefficient (wJC) and Correlation Coefficient (CC) for Artificial Data Sets

Figure 3. weighted Jaccard Coefficient wJC and Correlation Coefficient (CC) for Hybrid Data Sets

Figure 4. True and False Positives for $t6$

Figure 5. True and False Positives for $t5$

Figure 6. True and False Positives for $t4$

Figure 7. True and False Positives for $h6$

Figure 8. True and False Positives for $h5$

Figure 9. True and False Positives for $h4$

	t6			t5			t4			mean
	5cl	10cl	20cl	5cl	10cl	20cl	5cl	10cl	20cl	
ngas	0.83 (1)	1.00 (1)	1.00 (1)	0.82 (1)	0.91 (1)	1.00 (1)	0.79 (2)	0.89 (1)	0.95 (1)	0.91
kmeans	0.83 (2)	0.94 (3)	1.00 (2)	0.81 (3)	0.91 (2)	0.98 (3)	0.79 (1)	0.86 (2)	0.93 (2)	0.89
hardcl	0.80 (3)	1.00 (2)	1.00 (3)	0.81 (2)	0.87 (3)	0.99 (2)	0.67 (3)	0.81 (3)	0.90 (3)	0.87
som	0.63 (5)	0.82 (4)	0.90 (5)	0.63 (5)	0.81 (4)	0.88 (4)	0.61 (4)	0.81 (4)	0.87 (4)	0.77
maximin	0.80 (4)	0.80 (5)	0.74 (6)	0.75 (4)	0.75 (5)	0.75 (5)	0.35 (7)	0.64 (5)	0.61 (5)	0.69
clara	0.32 (8)	0.76 (6)	0.49 (8)	0.34 (8)	0.63 (6)	0.52 (6)	0.35 (8)	0.44 (7)	0.54 (6)	0.49
cmeans	0.39 (6)	0.44 (7)	0.51 (7)	0.44 (6)	0.46 (7)	0.48 (7)	0.47 (5)	0.49 (6)	0.51 (7)	0.47
ufcl	0.39 (7)	0.35 (8)	0.94 (4)	0.36 (7)	0.35 (8)	0.39 (8)	0.38 (6)	0.41 (8)	0.43 (8)	0.44

Table 1. CC Ranks for Artificial Data Sets

	h6			h5			h4			mean
	5cl	10cl	20cl	5cl	10cl	20cl	5cl	10cl	20cl	
ngas	0.49 (1)	1.00 (2)	1.00 (3)	0.46 (1)	1.00 (1)	1.00 (2)	0.45 (1)	0.51 (2)	1.00 (1)	0.77
kmeans	0.46 (2)	1.00 (1)	1.00 (1)	0.42 (2)	1.00 (2)	1.00 (1)	0.38 (4)	0.68 (1)	1.00 (2)	0.77
ufcl	0.30 (5)	0.97 (3)	0.97 (4)	0.17 (7)	0.96 (3)	0.96 (4)	0.18 (7)	0.40 (5)	0.41 (6)	0.59
hardcl	0.38 (3)	0.49 (5)	1.00 (2)	0.36 (4)	0.51 (5)	1.00 (3)	0.37 (5)	0.41 (4)	0.47 (5)	0.55
clara	0.27 (6)	0.82 (4)	0.78 (6)	0.28 (5)	0.74 (4)	0.72 (6)	0.30 (6)	0.16 (7)	0.26 (7)	0.48
som	0.25 (7)	0.48 (6)	0.80 (5)	0.25 (6)	0.44 (7)	0.75 (5)	0.25 (2)	0.38 (6)	0.66 (3)	0.47
cmeans	0.37 (4)	0.45 (7)	0.59 (7)	0.38 (3)	0.46 (6)	0.56 (7)	0.39 (3)	0.46 (3)	0.52 (4)	0.46
maximin	0.06 (8)	0.18 (8)	0.15 (8)	0.11 (8)	0.09 (8)	0.16 (8)	0.09 (8)	0.06 (8)	0.12 (8)	0.11

Table 2. CC Ranks for Hybrid Data Sets

	t6			t5			t4			mean
	5cl	10cl	20cl	5cl	10cl	20cl	5cl	10cl	20cl	
maximin	0.98 (1)	0.98 (1)	0.98 (1)	0.97 (1)	0.98 (1)	0.98 (1)	0.09 (8)	0.87 (1)	0.91 (1)	0.86
ngas	0.44 (2)	0.98 (3)	0.98 (2)	0.40 (2)	0.75 (3)	0.98 (2)	0.37 (1)	0.69 (2)	0.90 (2)	0.72
kmeans	0.43 (3)	0.84 (5)	0.98 (3)	0.39 (3)	0.73 (4)	0.95 (4)	0.37 (2)	0.67 (3)	0.88 (3)	0.69
hardcl	0.37 (4)	0.98 (4)	0.98 (4)	0.33 (5)	0.60 (5)	0.96 (3)	0.29 (5)	0.55 (6)	0.80 (4)	0.65
clara	0.37 (5)	0.98 (2)	0.61 (7)	0.37 (4)	0.78 (2)	0.65 (7)	0.36 (3)	0.61 (4)	0.62 (7)	0.59
cmeans	0.32 (6)	0.55 (6)	0.61 (6)	0.31 (6)	0.53 (6)	0.72 (6)	0.31 (4)	0.57 (5)	0.72 (6)	0.52
som	0.14 (7)	0.50 (7)	0.81 (5)	0.14 (7)	0.49 (7)	0.80 (5)	0.13 (6)	0.48 (7)	0.78 (5)	0.47
ufcl	0.10 (8)	0.04 (8)	0.98 (4)	0.10 (8)	0.04 (8)	0.01 (8)	0.09 (7)	0.04 (8)	0.01 (8)	0.15

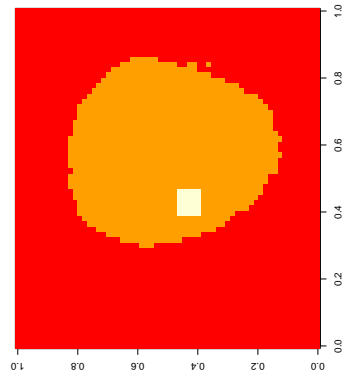
Table 3. wJC Ranks for Artificial Data Sets

	h6			h5			h4			mean
	5cl	10cl	20cl	5cl	10cl	20cl	5cl	10cl	20cl	
kmeans	0.30 (2)	0.96 (1)	0.96 (1)	0.25 (2)	0.96 (3)	0.96 (1)	0.23 (3)	0.63 (3)	0.96 (1)	0.69
ngas	0.25 (3)	0.96 (1)	0.96 (1)	0.24 (3)	0.96 (2)	0.96 (1)	0.24 (2)	0.30 (5)	0.96 (1)	0.65
cmeans	0.38 (1)	0.67 (2)	0.88 (3)	0.32 (1)	0.66 (4)	0.87 (4)	0.33 (1)	0.67 (2)	0.84 (3)	0.62
som	0.21 (4)	0.67 (3)	0.88 (4)	0.21 (4)	0.66 (5)	0.87 (5)	0.21 (4)	0.68 (1)	0.86 (2)	0.53
clara	0.16 (5)	0.96 (1)	0.96 (2)	0.16 (5)	0.96 (3)	0.95 (3)	0.16 (5)	0.31 (4)	0.45 (5)	0.56
ufcl	0.09 (7)	0.96 (1)	0.96 (1)	0.08 (7)	0.96 (1)	0.96 (1)	0.07 (7)	0.29 (6)	0.51 (4)	0.54
hardcl	0.10 (6)	0.28 (4)	0.96 (1)	0.10 (6)	0.39 (6)	0.96 (2)	0.09 (6)	0.14 (7)	0.19 (6)	0.36
maximin	0.02 (8)	0.04 (5)	0.05 (5)	0.02 (8)	0.04 (7)	0.06 (6)	0.02 (8)	0.04 (8)	0.05 (7)	0.04

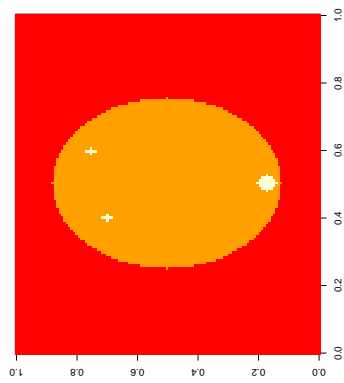
Table 4. wJC Ranks for Hybrid Data Sets

	t6	t5	t4	h6	h5	h4	mean
ward	0.98 (1)	0.98 (1)	0.73 (2)	0.96 (1)	0.89 (1)	0.74 (1)	0.88
complete	0.98 (2)	0.98 (2)	0.88 (1)	0.07 (2)	0.06 (2)	0.45 (2)	0.57
single	0.00 (3)	0.00 (3)	0.00 (3)	0.01 (3)	0.01 (3)	0.01 (3)	0.00
ward	1.00 (2)	1.00 (1)	0.74 (2)	1.00 (1)	0.84 (1)	0.81 (1)	0.90
complete	1.00 (1)	0.99 (2)	0.82 (1)	0.35 (2)	0.36 (2)	0.40 (2)	0.65
single	0.34 (3)	0.38 (3)	0.43 (3)	0.33 (3)	0.34 (3)	0.36 (3)	0.36

Table 5. wJC and CC Ranks for Hierarchical Methods



(b) Hybrid Set



(a) Artificial Set

