International Journal of Fuzzy Mathematics and Systems.
ISSN 2248-9940 Volume 13, Number 1 (2023), pp. 21-38

http://www.ripublication.com

Experiments using Minimax Data with Relational Fuzzy c-means and Assignment-Prototype Clustering Algorithms

Richard J. Hathaway[a], Timothy C. Havens[b] and Elizabeth D. Hathaway[c*]

[a]Department of Mathematical Sciences, Georgia Southern University,
1332 Southern Drive, Statesboro, GA 30458, United States
[b]Department of Computer Science, Michigan Technological University,
1400 Townsend Drive, Houghton, MI 49931, United States
E-mail: thavens@mtu.edu
[c]Department of Health and Human Performance,
University of Tennessee at Chattanooga, 615 McCallie Ave, Chattanooga, TN 37403
Email: elizabeth-hathaway@utc.edu
Phone: 423-425-5214; Fax: 423-425-4457 (corresponding author)

Abstract:

The visual assessment of (cluster) tendency (VAT) algorithm produces heatmaps that indicate cluster structure via dark diagonal blocks. While VAT is effective for well-separated, cloud-like clusters, it often does a poor job when clusters have shapes that are very different from clouds, such as chains, strings, lines, etc. The improved visual assessment of (cluster) tendency (iVAT) algorithm is much superior to VAT in most non-cloud cases and is able to make the improvement by changing from the original dissimilarity data to derived minimax dissimilarities. Relational fuzzy c-means (RFCM) and assignment-prototype (AP) clustering methods, like VAT, work well for well-separated clouds but fail in some non-cloud cases. We extend these algorithms using the same approach that improved VAT to iVAT; replacing the original dissimilarities with the minimax version. The new minimax-based clustering algorithms, fuzzy c-chains (FCC) and chain assignment prototype (CAP), are presented and results of some numerical experiments are given. In all the experiments our new methods are able to accurately capture chain clusters.

Keywords: Relational fuzzy c-means clustering, assignment-prototype clustering, improved visual assessment of cluster tendency, minimax dissimilarity

Acknowledgements:
This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

1. Introduction

The purpose of this paper is to present a minimax approach that will sometimes enhance the clustering capabilities of the *relational fuzzy c-means* (RFCM) [1] and the *assignment-prototype* (AP) clustering algorithms [2]. The RFCM algorithm has a strong duality relationship with the widely used fuzzy c-means (FCM) algorithm [3], and therefore our minimax approach can also be viewed as an improvement of FCM for certain data sets.

Clustering is the problem of partitioning a set of n objects $O = \{o_1, o_2, \ldots, o_n\}$ into c self-similar groups or clusters $C_1, \ldots, C_c$. Ideally, all objects in a computed cluster are similar to one other and any two objects from two different clusters are dissimilar. Clustering is typically done on the basis of numerical data, either object data or relational data. In the case of object data, O has a corresponding real $p \times n$ matrix X, whose k^{th} column $x_k \in \Re^p$ is called the feature vector for object o_k and gives numerical measurements of p features of o_k, such as, length, weight, etc. In the most straightforward case, clustering O amounts to looking for *compact and well-separated* groups of feature vectors in X.

In the case of relational data, O is described less directly through data arrayed in a real $n \times n$ matrix $R = [R_{ij}]$. In general, R_{ij} gives the relation between objects o_i and o_j, but in this paper, R will always be dissimilarity data satisfying

$$R_{ii} = 0, \forall\ i;\ R_{ij} = R_{ji}, \forall\ i, j;\ \text{and}\ R_{ij} \geq 0, \forall\ i, j. \tag{1}$$

Adding the triangle inequality $R_{ij} \leq R_{ik} + R_{kj}$ makes R a set of distance values, but this additional condition is not required for the methods considered here. In the case of dissimilarity data, our goal is to cluster the objects in O into groups of similar objects, as indicated by low dissimilarity values in R. While it is not generally possible to go from R to X, it is always easy to go from X to R. One often used conversion in the latter direction is to calculate the elements of the dissimilarity matrix R as pairwise squared-Euclidean (aka 2-norm) distances of the columns of X:

$$R_{SE_{ij}} = \left\|x_i - x_j\right\|_2^2 = \left(x_i - x_j\right)^T\left(x_i - x_j\right), \forall\ i, j. \tag{2}$$

The choice of R in (2) is special in the case of the FCM algorithm, which we will describe further in Section 3.

The optimization-based clustering methods presented in Section 3 produce fuzzy clusters, where each object o_k is allowed to have partial membership in several clusters. One advantage of a fuzzy clustering is that "in between" objects are easily identifiable in the clustering result; these objects have near equal membership in multiple clusters. A useful tool for representing a fuzzy clustering is called a fuzzy partition matrix. The set of all (nondegenerate) $c \times n$ fuzzy partition matrices of n objects $\{o_1, \ldots, o_n\}$ into c fuzzy clusters $C_1, \ldots, C_c$ is denoted by M_{fcn}, defined as

$$M_{fcn} = \{U \in \Re^{c\times n} \mid U_{ik} \in [0,1], \forall\ i,k;\ \textstyle\sum_{i=1}^{c} U_{ik} = 1, \forall\ k;\ \text{and}\ \sum_{k=1}^{n} U_{ik} > 0, \forall\ i\}. \tag{3}$$

We illustrate the proper interpretation of a partition matrix using the example

$$U = \begin{bmatrix} .25 & 1.0 & .05 & \boxed{.33} & .25 & .90 & .00 & \boxed{.40} & .25 & \boxed{.80} & .05 & .23 \\ .55 & 0.0 & .15 & \boxed{.34} & .75 & 0.0 & .10 & \boxed{.40} & .55 & .10 & .15 & .44 \\ .20 & 0.0 & .80 & \boxed{.33} & .00 & .10 & .80 & \boxed{.20} & .20 & .10 & .80 & .33 \end{bmatrix}, \quad (4)$$

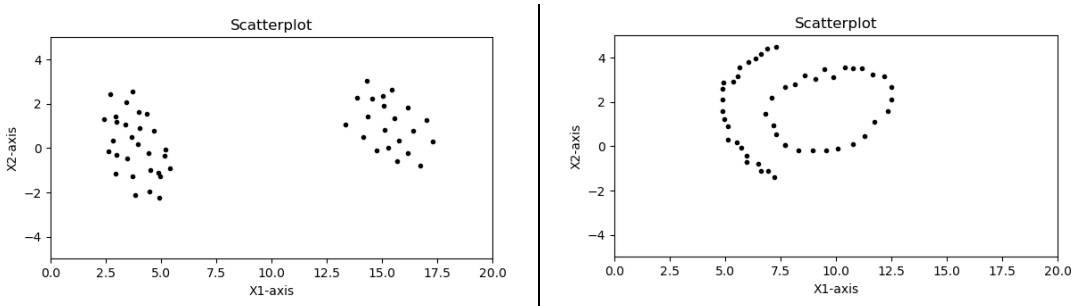
where the i^{th} row of U corresponds to cluster C_i and the k^{th} column of U corresponds to object o_k. We see that U in (4) is the representation of a fuzzy clustering of 12 objects into 3 clusters. The entry U_{ik} gives the degree to which object o_k belongs to (fuzzy) cluster C_i. These fuzzy membership values range from 0 (no membership) to 1 (complete membership). The sum of the memberships in each column is 1, which requires each object to be completely contained in the union $C_1 \cup C_2 \cup ... \cup C_c$. Notice that the value of U_{17} = 0.80 asserts that object o_7 has a high degree of membership in cluster C_1 and that the very close values of U_{14} =.33, U_{24} =.34, and $U_{3,4}$ =.33 indicate that object o_4 is an inlier or bridge object between all 3 clusters.

Fuzzy partitions provide detailed information (e.g., identifying inliers, etc.) but it is often useful to convert a fuzzy partitioning solution to a crisp one where each object is assigned total membership in exactly one of the clusters. This conversion of fuzzy to hard is usually done by placing an object in the cluster for which its fuzzy membership value is greatest. To illustrate an application of this "rule of maximum membership" we convert the fuzzy partition matrix in (4) to the crisp one

$$\widehat{U} = \begin{bmatrix} 0 & 1 & 0 & 00 & 1 & 0 & 10 & \boxed{1} & 0 & 0 \\ 1 & 0 & 0 & 11 & 0 & 0 & 01 & \boxed{0} & 0 & 1 \\ 0 & 0 & 1 & 00 & 0 & 1 & 00 & 0 & 1 & 0 \end{bmatrix}. \quad (5)$$

Notice that hardening U in (4) requires us to arbitrarily choose whether to place o_7 in C_1 or C_2; the hardened partition shown in (5) reflects the choice to put o_8 in C_1. Throughout the experimentation done in this paper, the rule of maximum membership is applied to produce the clustering labels used in labeled scatterplots and any assessment of clustering errors.

With the notation given, we now describe the topic considered here. Two major types of clusters are those having cloud structure and those having chain structure, each demonstrated with small 2-dimensional examples in Figure 1.



(a) 2 cloud clusters (b) 2 chain clusters

Figure 1. Illustration of two types of clusters

The FCM algorithm [3] is very effective for clouds but usually is not effective for partitioning a data set into chain clusters such as those in Figure 1(b). The most used form of FCM consists of doing a simple alternating optimization iteration (between U and V variables) that finds a minimizer (U*,V*) of the FCM function

$$J_2(U,V) = \sum_{i=1}^{c} \sum_{j=1}^{n} U_{ij}^2 \left\| x_j - V_i \right\|_2^2 \qquad (6)$$

where: c is the number of clusters assumed, n is the sample size of the object data set $X = [x_1, x_2, \ldots, x_n]$, variable U is a partition in M_{fcn}, and variable $V = [V_1, \ldots, V_c]$ is the matrix of cluster prototype vectors (or cluster means) which lie in the same space as the object data. It is not hard to see that $J_2(U,V)$ will be minimized for separated cloud clusters when U partitions the data into the various clouds and the prototypes V are (approximately) equal to the means of the data in each cloud.

Understand that the function $J_2(U,V)$ is naturally tuned to clouds. It represents the i^{th} cluster structure by its mean V_i, which is a reasonable cluster approximation for clouds but not for chains. Also, it uses the Euclidean norm (aka 2-norm) $\|\bullet\|_2$, whose unit ball is a hypersphere which describes a "perfectly" shaped cloud. Use of the Euclidean norm makes the function, and hence the clustering it produces, blind to any chain structure that exists in the data.

Previous research has been done to adapt the FCM approach to clusters with other types of structure. One approach changes the norm used in (6). Other L_p norms besides p = 2 are considered in [4] but that modification only slightly changes the type of cluster cloud sought. The method in [5] does allow better handling of elongated clusters using a weighted Euclidean distance but it still is effectively restricted to clouds. Another approach to extending FCM to other cluster structures involves changing the prototypes. Dave [6] considers circular shells for a specific application where circular and elliptical boundaries are being sought in digital images. Linear structures and convex combinations thereof are considered in [7]. There are two serious problems with the prototype variant approach for general data sets X: (1) this approach requires you to know what the cluster structure is (e.g., lines or ellipsoidal shells) so that you can try to use the appropriate prototype, and (2) the optimization of the corresponding function is typically much more difficult than optimizing J_2 in (6). Much of the early work on FCM variants is covered in [8], but the authors know of no earlier work that has successfully adapted an FCM approach to chain clusters.

Here we give two new relational fuzzy clustering methods. One of them is based on relational FCM clustering [1] and the other is based on assignment-prototype clustering [2]. Both methods are adapted to chain clusters and can be implemented using a combination of existing algorithms. In the case of FCM, our approach will recast the object data problem involving X to a relational data problem involving R_{SE} from (2). Then the squared-Euclidean R_{SE} will be converted to its *minimax* (mM) version R_{mM} using the computationally efficient approach given in [9]. This conversion will distort the Euclidean distances so that all objects in the same chain will now be nearby each other. Finally, the relational data R_{mM} will be clustered using a safeguarded relational data dual of FCM from [10]. In the case of the assignment-prototype method, we need do nothing more than apply its clustering algorithm to

R_{mM}. These two new methods, *fuzzy c-chains* (FCC) and *chain assignment-prototype* (CAP) will produce a fuzzy partition U in M_{fcn} of the original set of objects O.
In the next section we look at a small example to make the definition of minimax dissimilarity clear, and then state the algorithms used to efficiently obtain R_{mM}. Section 3 gives statements of FCC and CAP along with the earlier relational data clustering algorithms that are used in their definitions.The next section will give descriptions and results of several numerical experiments that are chosen to assess the performance and limitations of the new approaches. The final section will give a restatement of what is done here, our conclusions and a list of possible topics for future research related to this topic.

2 Calculating minimax dissimilarities

We begin our discussion of minimax dissimilarity using the following tiny 2-dimensional object data set

$$X = [x_1 \quad x_2 \quad x_3 \quad x_4] = \begin{bmatrix} 1 & 33 & 4 \\ 3 & 34 & 2 \end{bmatrix} \tag{7}$$

We convert X to *squared-Euclidean* (SE) relational data using (2) to get

$$R_{SE} = \begin{bmatrix} 0 & 4 & 5 & 10 \\ 4 & 0 & 1 & 2 \\ 5 & 1 & 0 & 5 \\ 10 & 2 & 5 & 0 \end{bmatrix} \tag{8}$$

and represent the relationship between objects using R_{SE} in the weighted graph in Figure 2.

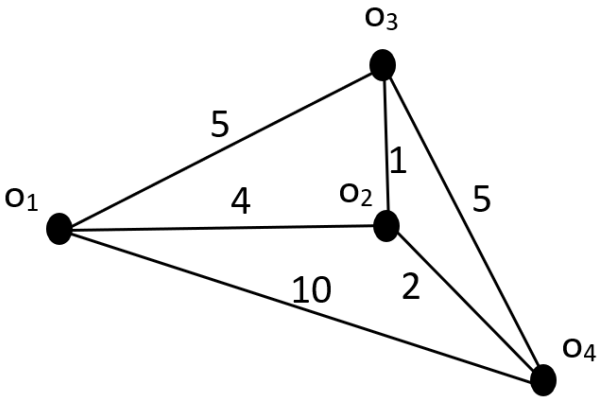


Figure 2. Representation of R_{SE} in (8) by weighted graph G.

To define the mM dissimilarity between a pair of distinct objects in G, all paths connecting the two vertices must be examined. For each path the largest edge length

is noted, and the mM dissimilarity is defined to be the minimum of those largest edge lengths. So, to find the mM dissimilarity between o_1 and o_4 we would look at the following paths and maximum edge lengths: path o_1, o_4 with max length = 10; path o_1, o_2, o_4 with max length = 4; path o_1, o_3, o_4 with max length = 5; and path o_1, o_2, o_3, o_4 with max length = 5. Since the minimum of the maximum path edge lengths is 4, we define the minimax dissimilarity between o_1 and o_4 to be 4.

It should be clear to the reader that the minimax dissimilarity will represent all objects corresponding to a chain as being similar and near to each other, and it is this property that should help any FCM or AP approach congregate all of a chain's points into the same cluster. The example is finished by giving the full mM matrix that is derived from R_{SE} in (8):

$$R_{mM} = \begin{bmatrix} 0 & 4 & 4 & 4 \\ 4 & 0 & 1 & 2 \\ 4 & 1 & 0 & 2 \\ 4 & 2 & 2 & 0 \end{bmatrix} \tag{9}$$

Note for a later discussion that transforming to mM data can shrink elements of the dissimilarity matrix, or leave them unchanged, but it cannot increase them.

For an efficient way to compute mM dissimilarities we examine two members of the *visual assessment of (cluster) tendency* (VAT) family of algorithms, which are nicely surveyed in [11]. The most basic of these algorithms is from [12] and is simply called VAT. The idea is to use the original R to reorder the objects starting with one on the outskirts of the data set, labeling it as $o_{(1)}$, then the next nearest object is added and designated $o_{(2)}$. The process continues, each time adding the next object from those not yet selected that is closest to any one of those that have been selected. This produces a permutation $o_{(1)}$, $o_{(2)}$, …, $o_{(n)}$ of the objects so that similar objects appear close to each other in this new ordering. Denoting the permutation by P, the reordered dissimilarity matrix R_P is easily obtained from R by applying P to similarly permute the rows and columns of R. After doing this reordering, VAT displays the reordered R_P as a heatmap where, in the case of a grayscale heatmap, small elements of R_P produce dark pixels and relatively large elements produce light pixels. In the case of well-separated cloud clusters, the image produced by R_P will indicate likely clusters as dark diagonal blocks on the main diagonal.

Similar to FCM, VAT is great for clouds but usually quite poor for chains. This problem is effectively addressed in [13] by converting the original dissimilarity data to mM dissimilarities. The result makes all objects in a given chain close to each other, but keeps objects in different chains distant. In almost all cases this *improved visual assessment of (cluster) tendency* (iVAT) method produces images superior to VAT images, as shown in an example later in this section (Fig. 3).

The original iVAT algorithm was further improved by the efficient implementation given in [9], which reduces the work required to compute R_{mM} from $O(n^3)$ to $O(n^2)$. Interestingly, the efficient implementation requires a VAT reordering to be done first. This means that both of the relational, optimization-based clustering techniques given

in the next section produce an iVAT image at no additional cost, and that image can provide help in specifying the number of clusters c and initial partition $U^{(0)}$.
Next, we give a statement of the two algorithms required to get the minimax dissimilarities.

Algorithm 1. VAT Reordering [12]

Input: R—n × n dissimilarity matrix

Step 1 Set J = {1, 2, …, n}, I = ∅; P(:) = (0, 0, …, 0).

Step 2 Select $(i,j) \in \underset{p\in J, q\in J}{\arg\max} \{R_{pq}\}$.
Set P(1) = i; I = {i}; and replace J ← J – {i}.

Step 3 For r = 2, …, n:
Select $(i,j) \in \underset{p\in I, q\in J}{\arg\min} \{R_{pq}\}$.
Set P(r) = j; replace I ← I ∪ {j} and J ← J – {i}.
next r.

Step 4 Obtain the ordered dissimilarity matrix R^* using the ordering array P as $R^*_{ij} = R_{P(i)P(j)}$ for $1 \le i, j \le n$.

Algorithm 2. Efficient iVAT Calculation of Minimax Dissimilarities [9]

Input: R^*—VAT reordered n × n dissimilarity matrix

Step 1 Initialize minimax dissimilarity matrix $R'^* = [0]^{n\times n}$

Step 2 For r = 2, …, n

j = $\underset{k=1,\ldots,r-1}{\arg\min} \{R^*_{rk}\}$

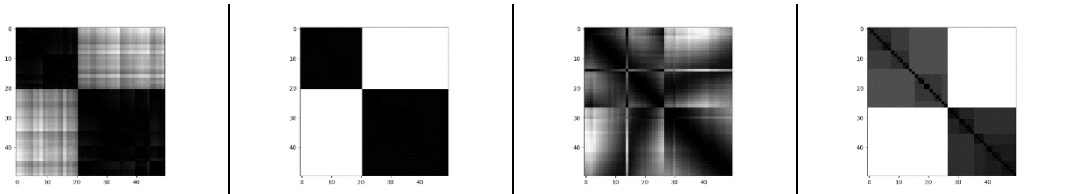
$R'^*_{rc} = R^*_{rc}$, c = j

$R'^*_{rc} = \max \{R^*_{rj}, R'^*_{jc}\}$, c = 1, …, r-1, c ≠ j

next r

Step 3 R'^* is symmetric, thus $R'^*_{rc} = R'^*_{cr}$

Programs for VAT and iVAT are widely available as open-source software online and a good Python implementation is available at [14]. We end this section by giving the VAT and iVAT images in Figure 3 for each of the two 50 point 2-dimensional data sets from Figure 1.



VAT for Figure 1(a) | iVAT for Figure 1(a) | VAT for Figure 1(b) | iVAT for Figure 1(b)

Figure 3.

Notice that the first two images in Figure 3 demonstrate that both VAT and iVAT can correctly assess 2 clusters in the case of well-separated clouds, but even in this very

easy problem, iVAT gives a much cleaner image than VAT. This difference in image quality is typical. For the chain clusters in Figure 1(b), there is a much greater difference between VAT and iVAT images. The image produced by VAT is somewhat difficult to properly interpret even though the two chains are quite well separated, while the iVAT image cleanly identifies the existence of 2 clusters. Notice the great improvement that mM data makes in the case of VAT and iVAT. Can the same data transformation help with optimization-based clustering algorithms such as FCM? In the next section we state the new methods that attempt to gain advantage for chain clusters by using mM dissimilarities.

3 Fuzzy c-chains and chain assignment-prototype clustering

We begin with the more challenging case of bringing minimax dissimilarities to FCM clustering. It is not obvious how to incorporate mM distances in place of the Euclidean ones in FCM function $J_2(U,V)$ in (6). Fortunately, there is a duality relationship enjoyed by FCM that allows us to recast any clustering in X with a clustering in R, using relational data R_{SE}. The relational dual of FCM is called the *relational fuzzy c-means* algorithm (RFCM) and is introduced in [1]. The duality is complete if the relational and object data satisfy (2) as the FCM and RFCM algorithms even produce the same sequence of partition iterates if initialized from the same starting partition $U^{(0)}$.

So, to do mM clustering with FCM, one might try the following approach: (1) convert X to R_{SE}; (2) convert R_{SE} to R_{mM} using iVAT; (3) cluster R_{mM} using RFCM. Unfortunately, there is a possible snag. The RFCM method is only guaranteed to work to completion if it is applied to relational data that are squared Euclidean; in some non-Euclidean cases it can fail, as signaled by calculation of negative "distances". It is not guaranteed that the R_{mM} values are actually squared-Euclidean distances for some (projected, but unknown) data set X, and so a failure to complete seems to be a possibility. We will salvage the relational approach by using the *non-Euclidean relational c-means* (NERFCM) algorithm to cluster the R_{mM} data produced by iVAT. It works by dynamically adding a number β to all off-diagonal elements of R_{mM} as needed to ensure that the core RFCM iteration is able to complete. Other strategies besides adding β to the off diagonals are explored in [15]. It is interesting to note that the conversion of R_{SE} to R_{mM} can decrease off-diagonal elements while NERFCM applied to R_{mM} can increase them. In the experiments of the next section, we will monitor the decreasing and increasing actions produced by the various steps of our procedure. Next, we state the NERFCM algorithm.

Algorithm 3. NERFCM Clustering Algorithm (with fuzzifier m = 2) [10]

Input: R—n × n dissimilarity matrix; $U^{(0)}$ —initial partition from set M_{fcn}. (Choice of $U^{(0)}$ also specifies c.)

Notation: R_β denotes the matrix obtained by adding β to each off-diagonal element of R;
$\|\bullet\|_2$ denotes the Euclidean (i.e., 2 norm) on $\Re^n$; and
e_k denotes the k^{th} unit vector in $\Re^n$.

Step 1 Initialize p = 0, β = 0, $R_0 = R$, and convergence tolerance ε.

Step 2 Calculate the new prototype vectors in $[V_1,\ldots,V_c] = V = V^{(p)}$ using $U = U^{(p)}$, for $1 \le i \le c$ with
$V_i = (U_{i1}^2, \ldots, U_{in}^2)/\left(\sum_{j=1}^{n} U_{ij}^2\right)$.

Step 3 Calculate the c×n matrix D of "squared" distances using $[V_1,\ldots,V_c] = V = V^{(p)}$, for $1 \le i \le c$, $1 \le k \le n$ with
$D_{ik} = \left(R_\beta V_i\right)_k - \left(V_i^T R_\beta V_i\right)$
If $D_{ik} < 0$ for any i and k, then:
calculate $\Delta\beta = \max\{-2D_{ik}/(\|V_i - e_k\|_2^2)\}$
update $D_{ik} \leftarrow D_{ik} + (\Delta\beta/2) * \|V_i - e_k\|_2^2$ for $1 \le i \le c$, $1 \le k \le n$
update $\beta \leftarrow \beta + \Delta\beta$

Step 4 Calculate the new partition matrix $U = U^{(p+1)}$ using squared distances D, for $1 \le k \le n$ with
If $D_{ik} > 0$ for all $1 \le i \le c$, then: $U_{ik} = (1/D_{ik})/\sum_{j=1}^{c}(1/D_{jk})$.
else, then: $U_{ik} = 0$ if $D_{ik} > 0$, $U_{ik} \in [0,1]$, $\sum_{j=1}^{c} U_{jk} = 1$

Step 5 If $\max_{ij}\left|U_{ij}^{(p)} - U_{ij}^{(p+1)}\right| < \varepsilon$, then: stop with $U^* = U^{(p+1)}$;
else, then: update p ← with p+1 and go to Step 2.

Now everything is in place to state our minimax-based fuzzy c-means approach to clustering.

Algorithm 4. FCC: Fuzzy c-Chains Clustering

Input: R—n × n dissimilarity matrix, either as original data or as calculated from a data set X;
$U^{(0)}$ —initial partition from set M_{fcn}.

Step 1 Apply Algorithm 2 (iVAT) to convert R to R_{mM}. (Choice for $U^{(0)}$ can also come from iVAT.)

Step 2 Apply Algorithm 3 (NERFCM) to cluster objects according to R_{mM}.

The adaptation of the *assignment-prototype* (AP) clustering algorithm in [2] is easy to describe. Similar to FCM, it consists of an alternating optimization of a particular clustering criterion function over the U and W variables, where $U \in M_{fcn}$ is the partition matrix (called the *assignment* matrix in [2]) and W is called the *prototype* weight matrix and is constrained to lie in

$$M_W = \{W \in \Re^{c\times n} \mid W_{ik} \in [0,1], \forall\, i,k;\ \textstyle\sum_{k=1}^{n} W_{ik} = 1, \forall\, i\} \qquad (10)$$

The AP algorithm is stated next, followed by its mM modification.

Algorithm 5. AP Clustering Algorithm [2]

Input: R—n × n dissimilarity matrix; $U^{(0)}$ —initial partition from set M_{fcn}. (Choice of $U^{(0)}$ also specifies c.)

Step 1 Initialize iteration number p = 0, and convergence tolerance ε.

Step 2 Calculate the new prototype weight matrix $W = W^{(p)}$ using $U = U^{(p)}$, for $1 \le i \le c$, $1 \le k \le n$ with
$W_{ik} = \left(1/\sum_{j=1}^{n} U_{ij}^2 R_{jk}\right) / \sum_{q=1}^{n} \left(1/\sum_{j=1}^{n} U_{ij}^2 R_{jq}\right)$.

Step 3 Calculate the new partition matrix $U = U^{(p+1)}$ using $W = W^{(p)}$, for $1 \le i \le c$, $1 \le k \le n$ with
$U_{ik} = \left(1/\sum_{j=1}^{n} W_{ij}^2 R_{jk}\right) / \sum_{q=1}^{c} \left(1/\sum_{j=1}^{n} W_{qj}^2 R_{jk}\right)$.

Step 4 If $\max_{ij} \left| U_{ij}^{(p)} - U_{ij}^{(p+1)} \right| < \varepsilon$, then: stop with $U^* = U^{(p+1)}$;
else, then: update p ← with p+1 and go to Step 2.

Algorithm 6. CAP: Chain AP Clustering

Input: R—n × n dissimilarity matrix, either as original data or as calculated from a data set X;
$U^{(0)}$ —initial partition from set M_{fcn}.

Step 1 Apply Algorithm 2 (iVAT) to convert R to R_{mM}. (Choice for $U^{(0)}$ can also come from iVAT.)

Step 2 Apply Algorithm 5 (AP) to cluster objects according to R_{mM}.

We close this section with several notes about the algorithms stated in this section. The AP and RFCM algorithms have been compared in earlier works [16], where they generally produced similar results, but those experiments did not involve mM derived data. While FCC and CAP can both handle any relational data satisfying (1), the strongest association between FCC and the original FCM algorithm occurs when it is initiated using squared-Euclidean data consistent with (2). A Python implementation of Algorithm 5 is available from the corresponding author and a MATLAB implementation of Algorithm 3 is available from [17].

Finally, it is most important to remember that the application of VAT (as part of iVAT) changes the ordering of the objects from that point on in the application of FCC or CAP, and therefore the permutation P produced by VAT should be taken into account when interpreting the clustering results. The user should either permute X using P to align the data (equivalently, the objects) with the VAT reordering of R or else permute the final partition U back to properly align it with the original ordering of X.

4 Experiments

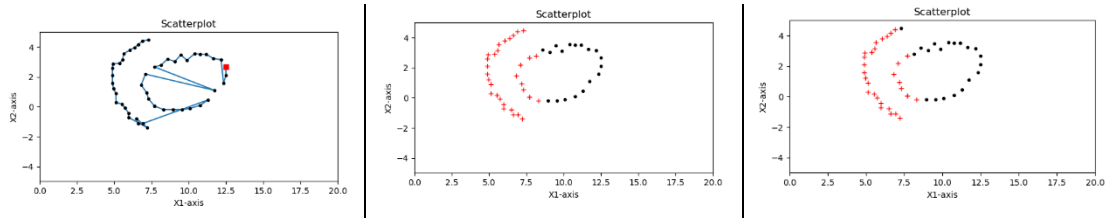
We begin with some general comments that will help interpret the results of all the experiments. In all but the last example we are looking at 2-dimensional examples so that we can compare the algorithm results with what our eyes see. Keep in mind that all these methods work on object data sets X of any dimensionality (by converting X to R) and even on pure relational data R that was not derived from object data. We use small examples in 2 dimensions to maximize what we can learn about the behavior of the methods.

Unless stated otherwise, all NERFCM and AP-based clusterings reported here were initialized using a hard partition consistent with the blocks in the corresponding iVAT image. This choice for initialization will have an ordered form similar to this example for c = 3 clusters and n = 12 objects,

$$U^{(0)} = \begin{bmatrix} 1 & 1 & 1\,0 & 0 & 0\,0 & 0 & 0\,0 & 0 & 0 \\ 0 & 0 & 0\,1 & 1 & 1\,1 & 1 & 0\,0 & 0 & 0 \\ 0 & 0 & 0\,0 & 0 & 0\,0 & 0 & 1\,1 & 1 & 1 \end{bmatrix}. \tag{11}$$

The ordering of X is always realigned with the ordering produced by VAT, which means that x_1 corresponds to the first datum chosen in the VAT ordering and x_n corresponds to the last chosen. We usually indicate the VAT ordering in scatterplots using edges, starting with a square marker for x_1.Using iVAT first, which is necessary in order to get the R_{mM} data needed by FCC and CAP, gives great $U^{(0)}$ initializations for applications of NERFCM and AP. Also, realigning X to be consistent with the VAT ordering makes it easier to interpret the terminal U partitions produced by NERFCM and AP, as nearby columns in U should have similar membership values up to the point where we jump from one cluster to another.

Example 1. We begin the experimentation by finishing our discussion of the data set of Figure 1(b). The data set in Figure 1(a) is unremarkable as it is easily clustered by NERFCM and AP, both using R_{SE} and R_{mM}. On the other hand, more happens with Figure 1(b) and we begin with another scatterplot that gives the VAT reordering of the 50 data points in Figure 4(a). Notice how VAT successfully exhausts one of the separated chains before hopping to the other. Figures 4(b) and 4(c) represent the maximum-membership partitions produced from the terminal partitions of NERFCM and AP (applied to S_{SE}), respectively. While the VAT ordering actually captures some of the cluster information, the optimization-based clustering schemes just search the data of squared-Euclidean distances for clouds. Notice there is only a slight difference between the NERFCM and AP results in Figure 4.



(a) VAT-ordered scatterplot (b) NERFCM R_{SE} clustering (c) AP R_{SE} clustering

Figure 4.

Now we consider the iVAT transformed data R_{mM}. In transforming from squared-Euclidean to mM data, the average off-diagonal element decrease for this example is approximately 16.2 and the sum of elements ratio,

$$\sum_{i=1}^{n}\sum_{j=1}^{n} R_{mM\,ij} / \sum_{i=1}^{n}\sum_{j=1}^{n} R_{SE\,ij}, \qquad (12)$$

is approximately 0.07, indicating a sizable reduction in off-diagonal elements. So how did FCC and CAP perform? Each was able to correctly capture the chains, as shown in Figure 5, where labels were obtained by maximum-membership hardening of terminal partitions.

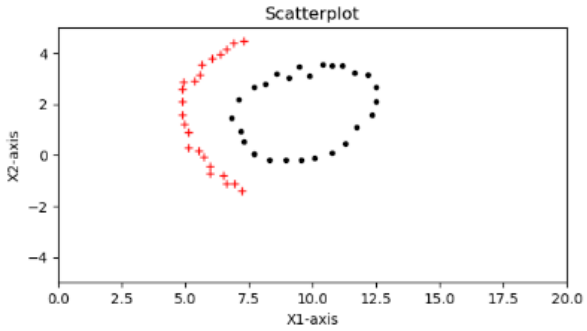


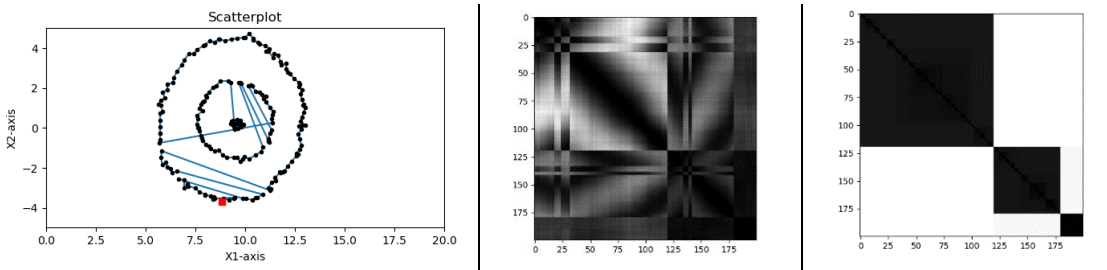
Figure 5. Clustering obtained by FCC and CAP

The results in Figure 5 indicate good success with the new approach, but there is a cost. The conversion from R_{SE} to R_{mM} causes a loss of information. In the case of cluster heat maps (i.e., VAT type images), the loss of information is often beneficial, giving cleaner images; but in our optimization-based clustering methods, the terminal partitions become slightly less informative. There are often many repeated element values in R_{mM} which leads to repeated values in the computed partition matrix U.For example, the portion of the terminal FCC partition corresponding to objects 24-31 is

$$U[1{:}2,\ 24{:}31] = \begin{bmatrix} .8684 & .8684 .8559 & .8559 .0875 & .0875 .0878 & .0895 \\ .1316 & .1316 .1441 & .1441 .9125 & .9125 .9122 & .9105 \end{bmatrix} \qquad (13)$$

Notice the frequently repeated columns in (13). Some ability to distinguish between objects may be diminished as the data become courser. While not considered here, we speculate that using a convex combination of R_{SE} with R_{mM} might lead to better overall performance by giving slightly more information in the terminal partitions. In additional experiments, we report that FCC and CAP were both able to converge to the visually-correct solution in Figure 5 even from poor initializations. No β adjustment was ever required when applying FCC.

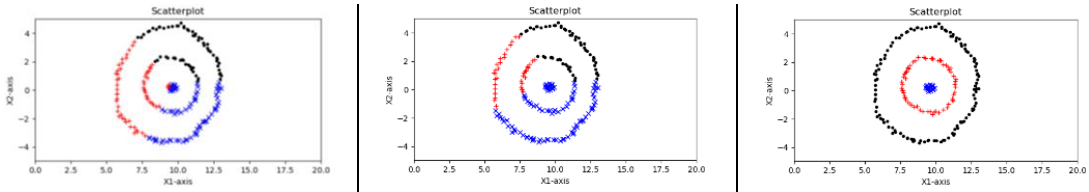
Example 2. We give a slightly more extreme case of the chain type structure seen in Example 1. The data set consists of 200 2-dimensional points arranged in a center cluster and 2 concentric bands. Figure 6 gives a scatterplot of the data indicating the VAT ordering, along with its VAT and iVAT images.



(a) VAT-ordered scatterplot (b) VAT image from R_{SE} (c) iVAT image from R_{mM}

Figure 6.

Notice the clear superiority of iVAT over VAT for indicating the presence of 3 clusters. The transformation from R_{SE} to R_{mM} resulted in an average off diagonal reduction of 19.2 and the value of the ratio in (12) is 0.05. The clustering results are illustrated in Figure 7, where view (c) shows the (visually correct) clustering that was obtained by both FCC and CAP.

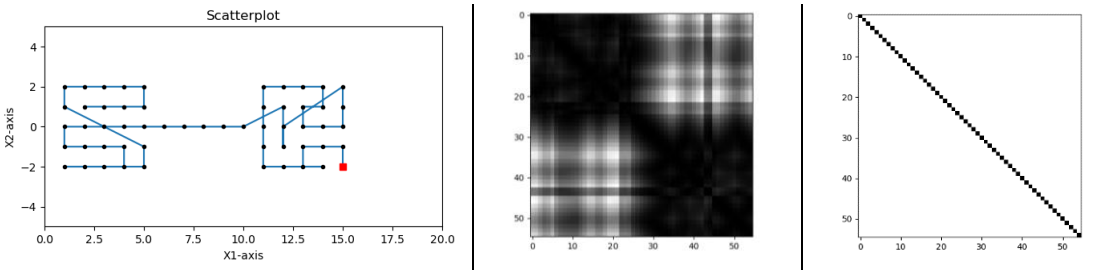


(a) NERFCM on R_{SE} (b) AP on R_{SE} (c) FCC and CAP (on R_{mM})

Figure 7.

The results in Figures 7(a-b) show that both NERFCM and AP fail to find the clusters but instead group cloudlike subsets of the data set, which is to be expected. However, use of the mM data in Figure 7(c) gives complete success. As in the previous example both FCC and CAP produced the clustering in Figure 7(c) from poor initializations without failure, and FCC never required a β adjustment.

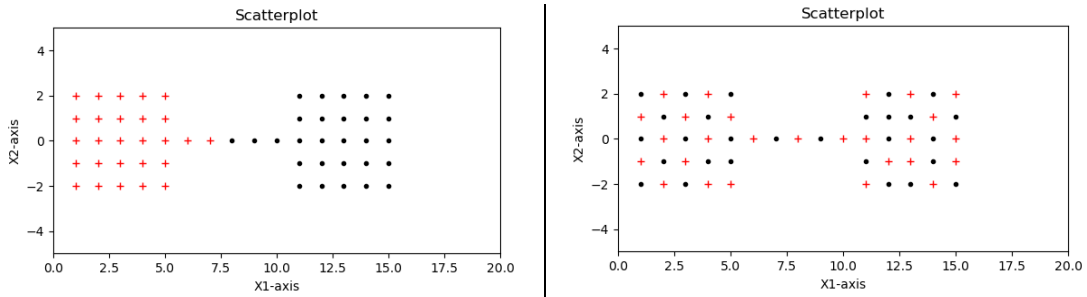
Example 3. The next example is a case where conversion to mM data destroys all dissimilarity information. The 55 points in Figure 8(a) are arranged in a regular grid with 25 points in each of the left and right boxes and 5 bridge points connecting the boxes. The Euclidean distance between each pair of horizontally or vertically adjacent points is 1. Using R_{SE} the VAT image in Figure 8(b) roughly captures the existence of 2 clusters with a small number of bridge points, but the iVAT image using R_{mM} in Figure 8(c) places each datum in its own cluster, and represents all points as being equidistant from each other (with all off-diagonal dissimilarities = 1).



(a) VAT-ordered scatterplot (b) VAT image from R_{SE} (c) iVAT image from R_{mM}

Figure 8.

Given the lack of information in Figure 8(c) we should expect nothing from FCC and CAP in this case. All clustering results of this experiment can be described using the two clusterings shown in Figure 9. Both NERFCM and AP (using R_{SE}) are able to find the good clustering in Figure 9(a), when initialized from either the good or poor initialization. On the other hand, both FCC and CAP (using R_{mM}) fail to move enough from the point of initialization to produce a terminal partition that hardens to a different clustering. In other words, initialize them at the good clustering and they produce the good clustering as hardened terminal partitions; initialize them with the poor clustering and they produce the same poor clustering as a result. As expected, no information in iVAT translates to no information for FCC and CAP. While neither algorithm moves enough from the point of initialization so as to produce a different hardened terminal partition, they do drive membership values of every point for every cluster to numbers very near or at 0.5. As with all prior experiments, there was no need to do any β spreading during the FCC iteration. The transformation from R_{SE} to R_{mM} resulted in an average off diagonal reduction of 53.1 and the value of the ratio in (12) is 0.02.



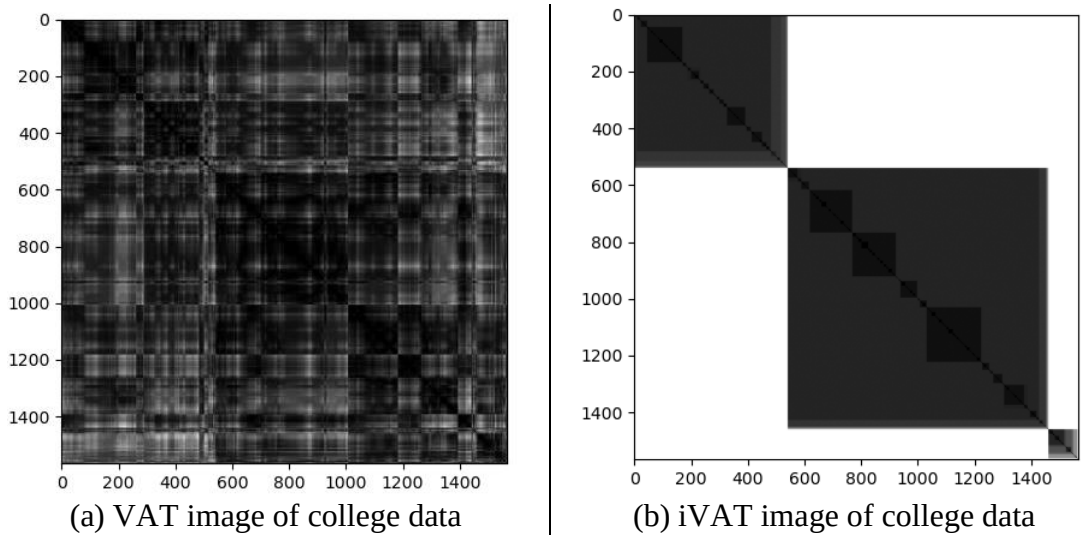
(a) Good clustering result/initialization (b) Poor clustering result/initialization

Figure 9.

Example 4. We conclude the experimentation using data from the 2013 College Scorecard accessible at data.world [18]. The purpose of the College Scorecard project is to provide data to help students and families compare college costs. The total data

set consists of 1,953 features with at least partial data listed for 7,804 colleges. To create our data set we chose parameter CURROPER = 1, which restricts the data screening to currently (in 2013) operating institutions, and PREDDEG = 3, which restricts the screening to predominantly undergraduate degree granting institutions. Data was then collected on 4 features: CCSIZSET (12 possible values; includes all types of 4-year institutions); CONTROL (3 possible values; includes public, private nonprofit, or private for profit); LOCALE (12 possible values; included all types of cities, suburbs, and rural areas); and REGION (8 possible values; excluding U.S. Service Schools and Outlying Areas). Any data with missing values from the above features were removed. The final screened data set consisted of 1,566 4-dimensional points (CCSIZSET, CONTROL, LOCALE, REGION). To provide equal weighting for each of the 4-dimensional features, all variables were transformed to a 1–12-point scale.

The VAT and iVAT images for our data set are shown in Figure 10 below. The iVAT image indicates c = 3 clusters in the data set. We initialized the optimization-based clustering methods using the initial partition indicated by the iVAT image with c = 3.



(a) VAT image of college data (b) iVAT image of college data

Figure 10.

So, what do the clusters mean in this case? Examination of the original feature values for data in each cluster tells us that the cluster membership split along CONTROL values (public, private nonprofit, or private for profit).Characteristics of cluster 1 (size = 542) include public university classification, 47% located in cities, 78% medium or large university classification, and 44% located in the Southeast and Mideast. Cluster 2 (size = 919) includes private nonprofit classification, 45% located in cities, 74% very small or small university classification, and 44% located in the Southeast and Mideast. Cluster 3 (size = 105) includes private for-profit classification, 60% located in cities, 80% very small or small university classification, and 50% of located in the Southeast and West.

The transformation from R_{SE} to R_{mM} resulted in an average off diagonal reduction of 55.3 and the value of the ratio in (12) is 0.15. The membership values produced by CAP were softer than those produced by FCC. As one example the x_{1080} membership values are $[0.063, 0.473, 0.464]^T$ and $[0.055, 0.869, 0.076]^T$ for CAP and FCC, respectively. Did the CAP and FCC iterations produce terminal partitions that were different from the good initialization based on the iVAT clustering? Changes occurred rarely for CAP with only 7 changes for the 1566 objects and never for FCC with 0 changes. Once again, no β adjustment was needed in any NERFCM calculations.

5 Conclusions

We have presented two new clustering approaches, FCC and CAP; they consist of first generating minimax dissimilarity data using iVAT and then applying either NERFCM for FCC or AP for CAP. Of the two optimization-based schemes, the FCC approach more often provided terminal partitions with less uncertainty that were closer to the hard partitions indicated by iVAT. Surprising to us–and frankly not yet understood–the FCC method never required any β adjustment. This means that FCC has, in our experience, always just been iVAT + RFCM, which gives a stronger relationship with the conventional object-data version of FCM [3]. Does minimax dissimilarity data ever need a β adjustment, and a slightly different question, are minimax dissimilarity data also squared-Euclidean data for some (unknown) object data set $\widehat{X}$?

The limited examples here give good evidence that FCC and CAP are superior to original c-means and assignment-prototype clustering in the case that the clusters are chains. Just as iVAT improves VAT for chain clusters, so does FCC and CAP improve NERFCM and AP, respectively. This is really to be expected since the minimax data forces all data in a chain cluster to be near each other.

Maybe the most obvious question to ask about this whole approach is why not just do iVAT, and use the iVAT identified clustering as the final clustering solution? In other words, what does post-processing of an iVAT solution by NERFCM (or possibly just RFCM) or AP add? It provides a fuzzy partition of the data, which: (1) can be used to identify inliers or otherwise provide more information about intra-and inter-cluster structure, (2) gives a way to identify exemplar objects for each of the clusters that might be useful in building a classifier, and (3) allows for the possibility of using additional cluster validity tools that are designed for fuzzy partitions. The truth is that doing NERFCM or AP beyond iVAT is a good idea if you want a fuzzy partitioning of the data that is consistent with the iVAT image; if not, then stop with iVAT. The *clustering in ordered dissimilarity* (CLODD) algorithm [19] is an existing method for finding crisp clusters from VAT/iVAT images.

We end with some questions that arose during this work: (1) Let R_{VAT} and R_{iVAT} be a pair of corresponding relational data matrices. What does the size of $\|R_{VAT} - R_{iVAT}\|$ tell us about the types and separations of cluster structures in the data? (2) With the same notation of (1), would a convex combination of R_{VAT} and R_{iVAT} ever improve on the use of R_{iVAT} alone in FCC or CAP? (3) Would it be useful to devise a condition number for relational data sets that could warn the user when slight errors in the data

could lead to big changes in clustering results, and how would that be done? Would it depend on how easily the problem becomes hard through the addition of bridge points? (4) Is there a theoretical result available stating when minimax dissimilarity data are also actually squared-Euclidean data? (5) Or aside from the previous question, is there a result showing that β adjustment is never necessary in FCC even if the minimax dissimilarities are not squared-Euclidean distances for some object data set? (6) Are there other ways that iVAT results can be used to usefully steer the outcome of an optimization-based clustering method? (7) How is the best way to hybridize cloud and chain clustering so that a single method handles both cases equally well?

References

[1] R.J. Hathaway, J.W. Davenport, J.C. Bezdek, Relational duals of the c-means algorithms, Pattern Recognition. 22 (1989) 205-212. http://doi.org/10.1016/0031-3203(89)90066-6.

[2] M.P. Windham, Numerical classification of proximity data with assignment measures, J. Classification. 2 (1985) 157-172. http://doi.org/10.1007/BF01908073.

[3] J.C. Bezdek, Pattern Recognition with Fuzzy Objective Function Algorithms. Plenum, New York (1981). http://doi.org/10.1007/978-1-4757-0450-1.

[4] R.J. Hathaway, J.C. Bezdek, Y. Hu, Generalized fuzzy c-means Clustering strategies using L_p norm distances, IEEE Transactions on Fuzzy Systems. 8 (2000) 576-582. http://doi.org/10.1109/91.873580.

[5] D.E. Gustaffson, W. Kessel, Fuzzy clustering with a fuzzy covariance matrix, Proc. IEEE–CDC. 2 (1979) 761–766. http://doi.org/10.1109/CDC.1978.268028.

[6] R.N. Dave, Generalized fuzzy c-shells clustering and detection of circular and elliptical boundaries, Pattern Recognition. 25 (1992) 713-721. https://doi.org/10.1016/0031-3203(92)90134-5.

[7] J.C. Bezdek, C. Coray, R. Gunderson, J. Watson, Detection and characterization of cluster substructure: II. Fuzzy c-varieties and convex combinations thereof, SIAM J. Appl. Math. 40 (1981), 358-372. https://doi.org/10.1137/0140030.

[8] M.S. Yang, A survey of fuzzy clustering, Mathematical and Computer Modelling. 18 (1993) 1-16. https://doi.org/10.1016/0895-7177(93)90202-A.

[9] T.C. Havens, J.C. Bezdek, An efficient formulation of the improved visual assessment of cluster tendency (iVAT) algorithm, IEEE TKDE. 23 (2011) 568-584. http://doi.org/10.1109/TKDE.2011.33.

[10] R.J. Hathaway, J.C. Bezdek, NERF c-means: non-Euclidean relational fuzzy clustering, Pattern Recognition. 27 (1994) 429-437. https://doi.org/10.1016/0031-3203(94)90119-8.

[11] D. Kumar, J.C. Bezdek, Visual approaches for exploratory data analysis: A survey of the visual assessment of clustering tendency (VAT) family of algorithms, IEEE SMC Magazine. 6 (2020) 10-48. http://doi.org/10.1109/MSMC.2019.2961163.

[12] J.C. Bezdek, R.J. Hathaway, VAT: A tool for visual assessment of (cluster) tendency, Proc. IJCNN, IEEE Press. (2002) 2225-2230. http://doi.org/10.1109/IJCNN.2002.1007487.

[13] L. Wang, T. Nguyen, J. Bezdek, C. Leckie, K. Ramamohanarao, iVAT and aVAT: enhanced visual analysis for cluster tendency assessment, in: M.J. Zaki, J.X. Yu, B. Ravindran, V. Pudi (Eds.), Advances in Knowledge Discovery and Data Mining. PAKDD 2010. Lecture Notes in Computer Science, 6118. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-13657-3_5.

[14] I.Lachheb, pyclustertend (Version 1.6.2) (accessed June 1, 2021). https://pypi.org/project/pyclustertend/.

[15] M. Khalilia, J. Bezdek, M. Popescu, J.M. Keller, Improvements to the relational fuzzy c-means clustering algorithm, Pattern Recognition. 47 (2014) 3920-3930. https://doi.org/10.1016/j.patcog.2014.06.021.

[16] J.C. Bezdek, R.J. Hathaway, M.P. Windham, Numerical Comparison of the RFCM and AP Algorithms for Clustering Relational Data, Pattern Recognition. 24 (1991) 783-791.

[17] M. Khalilia, NERFCM (accessed June 1, 2021). https://github.com/mohammedkhalilia/NERFCM

[18] U.S. Department of Education, College Scorecard, Data World Cluster Exercises (accessed July 28, 2021). https://data.world/exercises/cluster-analysis-exercise-2/workspace/file?filename=CollegeScorecard.csv.

[19] T.C. Havens, J.C. Bezdek, J.M. Keller, M. Popescu, Clustering in ordered dissimilarity data. Int. J. Intelligent Systems. 24 (2009) 504-528. https://doi.org/10.1002/int.20344.