A New Tree Clustering Algorithm for Fuzzy Data Based on α-cuts

Mohammad GhasemiGol, Mostafa Sabzekar, Reza Monsefi, Hadi Sadoghi Yazdi
Department of Computer Engineering, Ferdowsi University of Mashhad (FUM), Mashhad, Iran

***Abstract* -** *This paper presents a new approach to clustering fuzzy data, called Extensional Tree (ET) clustering algorithm by defining a dendrogram over fuzzy data and using a new metric between fuzzy numbers based on α-cuts. All the similar previous methods extended FCM to support fuzzy data. The present work is based on hierarchical clustering algorithm to cluster fuzzy data. In this novel approach a dendrogram is drawn over fuzzy or crisp data and then the desired clusters are extracted. Finally we compare this approach with some of the newly presented methods in the literature. The major advantage of ET is its fault tolerance against noisy samples. The overall experiments show prominence of our proposed method in comparison with other presented works.*

Keywords: Hierarchical Clustering method, Fuzzy data, Fuzzy dendrogram, Dissimilarity measure, α-cut

1 Introduction

Clustering methods are widely used in many scientific fields such as:- image processing, pattern recognition, data mining, machine learning, geology etc. Although there are many clustering algorithms, only few of them can cluster fuzzy data beside crisp data. For example the proposed algorithms [5,7,8,13] just work on crisp data. Fuzzy data is another type of data that is imprecise or with a source of uncertainty. This data type has been extensively used in natural language, social science, knowledge representation etc., due to its closeness with our way of life. Therefore we need clustering algorithms that can support this kind of data. Recently some methods have been presented in this field. Most of these algorithms extended FCM (that is the one of the widely used fuzzy clustering models [3, 9]) to cluster fuzzy data [4, 12, 14]. Hathaway et al. proposed FCM for fuzzy data. They defined a dissimilarity measure for two symmetric Trapezoidal Fuzzy Numbers (TFNs) and used it for FCM clustering [12]. In another example Miin-Shen Yang et al. presented Fuzzy clustering algorithms for mixed feature variables [4]. As an important deficiency, all of these algorithms cannot tolerate noise on samples. The proposed algorithm (ET) can resolve this defect by discovering noisy samples and clustering them into separate clusters.

Due to the usage of hierarchical method, the proposed algorithm has inherited all the benefits of this method, since it is very illustrative and clear that we can guesstimate dispersion of clusters in a glance at the dendrogram as well as its fault tolerance against noisy data.

This paper is organized as follows. In the next section we review hierarchical clustering method. The proposed distance and the new clustering algorithm (ET) are explained in section 3 & 4. In section 5 we focused on noisy samples. In the last section we evaluate the performance of ET and compare it to other similar proposed algorithms.

2 Review of hierarchical clustering method

Hierarchical clustering procedures are the most commonly used method of summarizing data structure. A hierarchical tree is a nested set of partitions represented by a tree diagram or dendrogram (see Fig. 1). Sectioning a tree at a particular level produces a partition into g disjoint groups. If two groups are chosen from different partitions (the results of partitioning at different levels) then either the groups are disjoint or one group wholly contains the other [1].

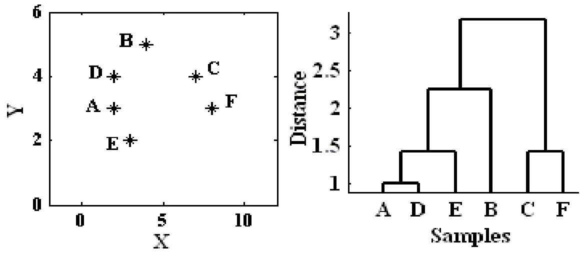

Fig. 1. Dendrogram is drawn over six samples.

The hierarchical algorithm contained the following procedure, where c is the desired number of final clusters. If c=1 then the dendrogram could be created [2].

Algorithm (Agglomerative hierarchical clustering)

- *Begin*
- *Initialize* $c, \hat{c} \leftarrow n, D_i \leftarrow \{x_i\}$, *i = 1,..., n*
- *Do* $\hat{c} \leftarrow \hat{c} - 1$
- *Find nearest clusters, say, Di and Dj*
- *Merge Di and Dj*
- *Until* $c = \hat{c}$
- *Return c clusters*
- *End*

Here c=1. By this technique we will draw cluster's dendrogram and use it to specify clusters. So at first, dissimilarity matrix is created. This matrix shows distances between each pair of samples. Suppose that at the beginning, every sample is a cluster with one sample. Then in each step two clusters that are closer to each other get selected and joined as a new cluster. At the end, we have a nested set of clusters that can be analyzed.

In the hierarchical method we use several mechanisms to obtain the distance of two clusters. One of which is single-link method. In this method the distance between two clusters is defined as the distance between their closest members of two clusters. In other words the distance between two groups, *A* and *B*, is defined as:

$$d_{AB} = \min_{i \in A, j \in B} (d_{ij})$$

Another mechanism is complete-link. In this method the distance of two clusters is defined as the distance between their furthest members of two clusters, i.e., the distance between two groups, A and B, is

$$d_{AB} = \max_{i \in A, j \in B} (d_{ij})$$

In this method, we make sure that other samples of two clusters are closer than the distance between them.

3 The proposed dissimilarity measure based on α-cuts

In this section, we consider the fuzzy data definition and explain the new method to obtain the distance between two fuzzy numbers. We define fuzzy data based on Hathaway's parametric model. We can extend symmetric trapezoidal fuzzy numbers (TFNs) to all TFNs by defining its parameterization as shown in Fig. 2. Parameterization of a trapezoidal fuzzy number A is denoted by $A = m(a_1, a_2, a_3, a_4)$ where a_1, a_2, a_3, a_4 are called center, inner diameter, left outer radius and right outer radius respectively [4] [15].

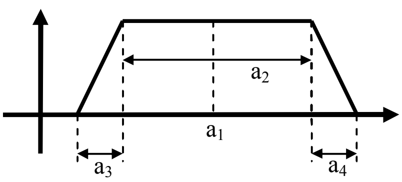

Fig. 2. Parameterization of trapezoidal fuzzy data.

The benefit of this representation is that we can easily show four kinds of fuzzy data (see Fig. 3). According to the representation above, A = [a_1, 0, 0, 0], B = [b_1, b_2, b_3, b_4], C = [c_1, c_2, 0, 0], D= [d_1, 0, d_3, d_4].

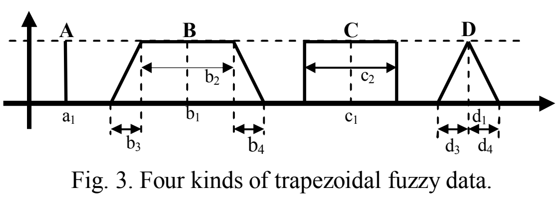

Fig. 3. Four kinds of trapezoidal fuzzy data.

Let $A = m(a_1, a_2, a_3, a_4)$ and $B = m(b_1, b_2, b_3, b_4)$ be any two fuzzy data. Hathaway et al. [12] defined dissimilarity for two TFNs A and B as follows:

$$d_h^2(A,B) = (a_1 - b_1)^2 + (a_2 - b_2)^2 + (a_3 - b_3)^2 + (a_4 - b_4)^2.$$

However, they did not consider the left or right shapes of numbers (i.e. LR- type TFN). Yang et al. [4] gave a distance for two symmetric TFNs A and B based on yang and Ko's distance definition [15] as follows:

$$d_f^2(A,B) = \frac{1}{4}\left(g_-^2 + g_+^2 + (g_- - (a_3 - b_3))^2 + (g_+ - (a_4 - b_4))^2 \right)$$

Where $g_- = 2(a_1 - b_1) - (a_2 - b_2)$

And $g_+ = 2(a_1 - b_1) + (a_2 - b_2)$

Here, we present a new method to obtain the distance between two trapezoidal fuzzy numbers that is based on α-cuts.

Definition 3.1: A fuzzy number (or interval) u is completely determined by any pair $u = (\underline{u}, \overline{u})$ of functions $\underline{u}, \overline{u} : [0,1] \to R$, defining the end-points of the α -cuts, satisfying the three conditions [10]:

- $\underline{u}(\alpha)$ is a bounded monotonic increasing (non-decreasing) left-continuous function for all $\alpha \in (0,1]$ and right-continuous for $\alpha = 0$.
- $\overline{u}(\alpha)$ is a bounded monotonic decreasing (non-increasing) left-continuous function for all $\alpha \in (0,1]$ and right-continuous for $\alpha = 0$.
- For all $\alpha \in (0,1]$ we have: $\underline{u}(\alpha) \le \overline{u}(\alpha)$ The notation

  $$u(\alpha) = [\underline{u}(\alpha), \overline{u}(\alpha)] \quad , \alpha \in [0,1]$$

  explicitly denoted the α -cuts of u. We refer to $\underline{u}$ and $\overline{u}$ as the lower and upper branches on u, respectively.

Definition 3.2: Let L (and R) be decreasing, shape functions from R^+ to [0, 1] with L(0)=1, L(x)<1 for all x>0, L(x)>0 for all x < 1, L(1)=0 or (L(x)>0 for all x and L(+∞)=0). A fuzzy number X with its membership function

$$\mu_X(x) = \begin{cases} L\left(\frac{m_1 - x}{\alpha}\right) & x \le m_1 \\ 1 & m_1 \le x \le m_2 \\ R\left(\frac{x - m_2}{\beta}\right) & x \ge m_2 \end{cases}$$

is called an LR-type TFN where $\alpha > 0$, $\beta > 0$ are called the left and right spreads respectively. Symbolically, X is denoted by $X = (m_1, m_2, \alpha, \beta)_{LR}$. A fuzzy number of LR-type, $u = (m_1, m_2, \gamma, \beta)_{LR}$, has α -cuts as follows:

$$u(\alpha) = [m_1 - \gamma L^{-1}(\alpha), m_2 + \beta R^{-1}(\alpha)]. \quad (1)$$

Definition 3.3: On the other hand the distance between two arbitrary fuzzy numbers $u = (\underline{u}, \overline{u})$ and $v = (\underline{v}, \overline{v})$ is defined as follows [6]:

$$d^2(u,v) = \int_0^1 (\underline{u}(\alpha) - \underline{v}(\alpha))^2 \, d\alpha + \int_0^1 (\overline{u}(\alpha) - \overline{v}(\alpha))^2 \, d\alpha. \quad (2)$$

Definition 3.4: The notation for the parameterization of a TFN A is $A = m(a_1, a_2, a_3, a_4)$ where we refer to a_1 as the center, a_2 as the inner diameter, a_3 as the left outer radius and a_4 as the right outer radius.

Thus, according to LR-type TFN representation, A is denoted by

$$A = (a_1 - \frac{a_2}{2}, a_1 + \frac{a_2}{2}, a_3, a_4)_{LR}, \quad (3)$$

And also since L and R are linear $(L(x) = R(x) = 1 - x)$ from Eq. (1) and Eq. (3) we obtain,

$$A(\alpha) = [\frac{2a_1 - a_2}{2} - (1-\alpha)a_3, \frac{2a_1 + a_2}{2} + (1-\alpha)a_4] \quad (4)$$

Thus, for any given two TFNs $A = m\,(a_1, a_2, a_3, a_4)$ and $B = m\,(b_1, b_2, b_3, b_4)$ we have the following distance $\delta_f^2(A,B)$, by Eq. (2) and Eq. (4) as follows:

$$\delta_f^2(A,B) = \int_0^1 \left(\frac{2a_1 - a_2}{2} - (1-\alpha)a_3 - \frac{2b_1 - b_2}{2} + (1-\alpha)b_3 \right)^2 d\alpha$$
$$+ \int_0^1 \left(\frac{2a_1 + a_2}{2} + (1-\alpha)a_4 - \frac{2b_1 + b_2}{2} - (1-\alpha)b_4 \right)^2 d\alpha.$$

4 The proposed extensional tree (ET) clustering algorithm

Now, we explain our proposed algorithm for clustering fuzzy data. This algorithm has five steps that will be explained in this section.

The Proposed Clustering Algorithm (ET)

- *Begin*
- *Initialize* $x = \begin{bmatrix} x_{11} & x_{12} & x_{13} & x_{14} \\ . & & & \\ . & & & \\ . & & & \\ x_{n1} & x_{n2} & x_{n3} & x_{n4} \end{bmatrix}$,

  (n=number of samples, each row of x identifies a fuzzy data).
- *Compute the distance between each pair of fuzzy data (see section 3).*
- *Create dissimilarity matrix.*
- *Create Fuzzy dendrogram.*
- *Extract clusters from fuzzy dendrogram (with inconsistency coefficient or max number of clusters).*
- *End*

Step1: For each fuzzy data, we need four crisp numbers to show it. So with x (is an n-by-4 matrix) we can present n fuzzy data samples.

Here we explain the ET algorithm to cluster fuzzy data by a simple example. Suppose that we have six fuzzy data (see Fig. 4).
A = [2.5; 2; 0.5; 1], B = [2.5; 0; 1; 1],
C = [7; 1; 0.5; 1], D = [8; 0; 1; 1],
E = [8.5; 0; 0.5; 0.5], F = [4.5; 0; 1; 1].

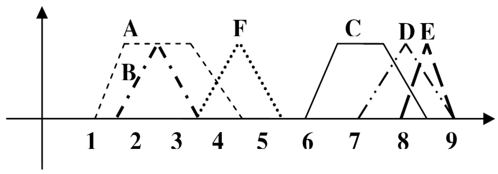

Fig. 4. Six fuzzy data A, B, C, D and E.

So, x matrix is shown as below.

$$x = \begin{bmatrix} 2.5 & 2 & 0.5 & 1 \\ 2.5 & 0 & 1 & 1 \\ 7 & 1 & 0.5 & 1 \\ 8 & 0 & 1 & 1 \\ 8.5 & 0 & 0.5 & 0.5 \\ 4.5 & 0 & 1 & 1 \end{bmatrix}$$

Step2: In this stage, we use our new formula to reach the distance between each two fuzzy data samples (For example $d_f(A,B)$=1.26, $d_f(A,C)$=6.40, …).

Step3: Now, we can easily create dissimilarity matrix. In this matrix each item shows the distance between two fuzzy data. This matrix is used for creating fuzzy dendrogram.

Table 1: Dissimilarity matrix for the samples in Fig. 4.

	A	**B**	**C**	**D**	**E**	**F**
A	0	1.26	6.40	7.70	8.46	2.93
B		0	6.56	7.78	8.49	2.83
C			0	1.35	2.14	3.75
D				0	***0.82***	4.95
E					0	5.67
F						0

Step4: Here, we should choose a method such as single-link, complete-link, median, etc. to create fuzzy dendrogram and form our clusters. We discussed two methods of them (single-link and complete-link) in section 2. Each method has its priority and can be useful in some cases.

- 'single-link ' --- nearest distance
- 'complete-link ' --- furthest distance
- 'average' --- unweighted average distance (UPGMA) (also known as group average)
- 'weighted' --- weighted average distance (WPGMA)
- 'centroid' --- unweighted center of mass distance (UPGMC)
- 'median' --- weighted center of mass distance (WPGMC)
- 'ward' --- inner squared distance (minimum variance algorithm)

If we run single-link method on the dissimilarity matrix that is shown in Table 1, we reach these results.

Table 2: The stages (a, b, c, d) of executing single-link method on Table 1.

Table 2.a

	A	B	C	{D,E}	F
A	0	***1.26***	6.40	7.70	2.93
B		0	6.56	7.78	2.83
C			0	1.35	3.75
{D,E}				0	4.95
F					0

Table 2.b

	{A, B}	C	{D,E}	F
{A,B}	0	6.40	7.70	2.83
C		0	***1.35***	3.75
{D,E}			0	4.95
F				0

Table 2.c

	{A, B}	{C,D,E}	F
{A, B}	0	6.40	***2.83***
{C,D,E}		0	3.75
F			0

Table 2.d

	{A,B,F}	{C,D,E}
{A,B,F}	0	***3.75***
{C,D,E}		0

You can see the produced dendrogram in this case, in Fig. 5.

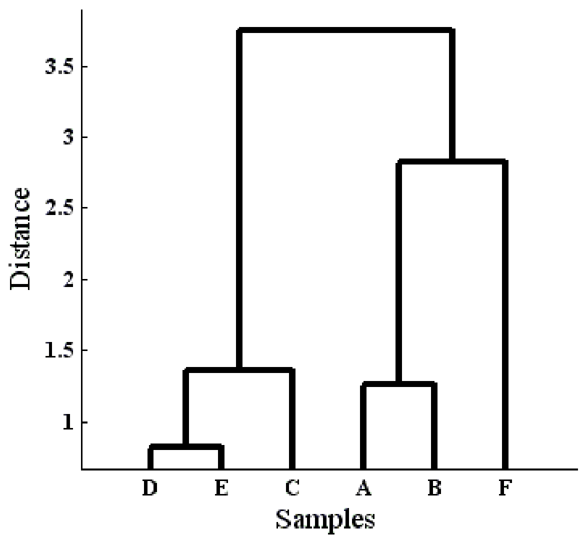

Fig. 5. Fuzzy dendrogram for the example.

Step5: This is the most important stage in our algorithm. Here we should extract the clusters from the fuzzy dendrogram. For this purpose there are two solutions. The first solution is based on the max number of clusters. For example if we want to have two clusters, it is enough to aggregate samples that are in each sub tree of dendrogram root, in a new cluster. In other words, we can divide dendrogram from the highest level into two clusters. This result is deducible from Fig. 4.

The other solution is based on the inconsistency coefficient [16]. In this way we define a label for each link in our dendrogram. This label shows how much two clusters are similar. With this measure, we can join clusters if the inconsistency value is less than specific threshold. The inconsistency coefficient characterizes each link in a cluster tree by comparing its length with the average length of other links at the same level of the dendrogram. The higher the value of this coefficient, the less similar the clusters connected by the link. To calculate inconsistency coefficient we should define two matrixes.

Linkage matrix is an (n-1)-by-3 matrix containing cluster tree information. The value of Linkage matrix for Fig. 4 is shown below.

Table 3: Linkage matrix for the samples in Fig. 4.

D	**E**	0.82
A	**B**	1.26
C	**{D,E}**	1.35
{A,B}	**F**	2.83
{A,B,F}	**{C,D,E}**	3.75

The other matrix is Inconsistency matrix that is an (n-1)-by-4, formatted as follows.

Table 4: Inconsistency matrix properties

Column	Description
1	Mean of the lengths of all the links included in the calculation.
2	Standard deviation of all the links included in the calculation.
3	Number of links included in the calculation.
4	Inconsistency coefficient.

We used the distance between two clusters, as length of the link that connects them to each other. If we name Linkage matrix, Z and Inconsistency matrix, W then the inconsistency coefficient for each link, is calculated by the following formula:

$$\mathbf{W(k,4) = (Z(k,3) - W(k,1)) / W(k,2)} \qquad (5)$$

For leaf nodes, nodes that have no further nodes under them, the inconsistency coefficient is set to zero.

Table 5: Inconsistency matrix for our example.

0.82	0	1	0
1.26	0	1	0
1.08	0.38	2	0.71
2.04	1.11	2	0.71
2.00	1.24	5	1.41

To understand this matrix, pay attention to the last row of inconsistency matrix. The calculation consists of five links, so the value of third column is five. Also the mean of used links is:

$$\mu = \frac{0.82 + 1.26 + 1.35 + 2.83 + 3.75}{5} = 2.00$$

And calculated Standard deviation is 1.24. For computing inconsistency coefficient we used formula (5).

$$\begin{aligned} W(5,4) &= (Z(5,3) - W(5,1)) / W(5,2) \\ &= (3.75-2.00) / 1.24 = 1.41 \end{aligned}$$

In the following figure, you can see the inconsistency coefficient related to each link.

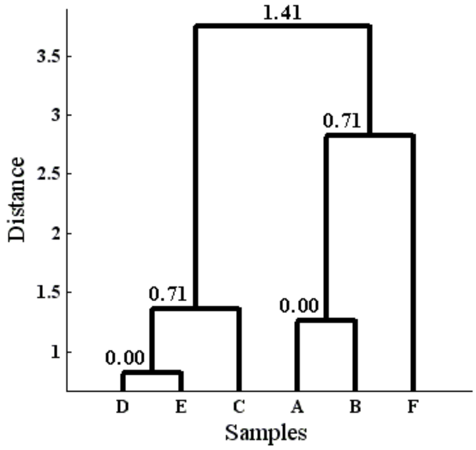

Fig. 6. Inconsistency coefficient related to each link.

Now, we can use inconsistency coefficient to form clusters. If we define threshold=1, then we have two clusters C1={A, B, F} and C2={C, D, E}.

According to Fig. 4 this result is perfectly acceptable. Therefore, in this method the number of clusters is defined via inconsistency coefficient and we don't have to define it.

5 Experimental results

In this section, we present an example that is shown the performance of the ET algorithm in comparison with the similar algorithms. An important benefit of this algorithm is its fault tolerance against noisy samples. We have tree structure of clusters and by analyzing that exactly, we can discover the noisy samples in two cases, low SNR[1] and medium SNR.

Case I: Low SNR

Indisputably, noisy samples are far from other samples, so the inconsistency coefficient between noisy samples and normal samples is larger. Thus, we can find noisy samples when the inconsistency coefficient grows up suddenly. In other words, this algorithm, noisy samples are classified in separated clusters. Usually, noisy samples appear in highest level of dendrogram.

Case II: Medium Level of SNR

ET algorithm can tolerate the medium domain of noise on some samples. For example if we have a little noise on a sample, it shouldn't have any effect on our clustering. In the other words, medium level of noise must not effect over clusters or clustering procedure must be robust against medium level of noise.

We consider a data set with six triangular fuzzy data (Fig. 7). We cluster them and then put a bit noise on a sample. Afterwards we cluster them again and compare the obtained results in two stages (Low level of noise on some samples). After clustering these samples, we obtained a dendrogram that is shown in Fig. 8.

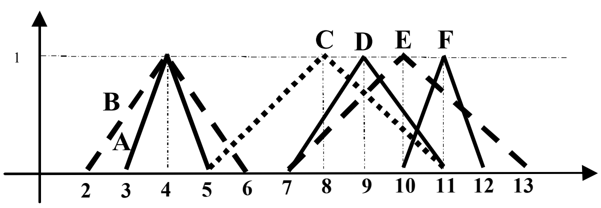

Fig. 7. Representation of six fuzzy data.

[1] Signal to Noise Ratio

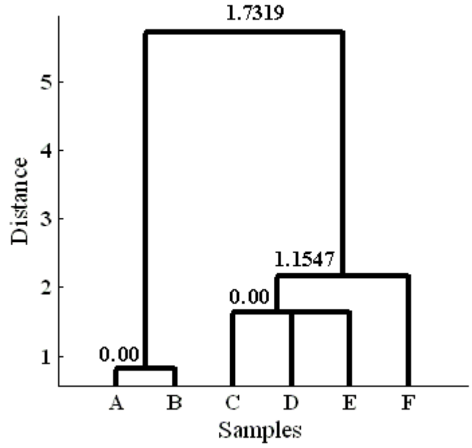

Fig. 8. Fuzzy dendrogram for the samples in Fig. 7.

As it can be seen, we have two clusters by applying the inconsistency coefficient and supposing threshold between 1.1547 and 1.7319 for example mean of them (threshold=1.4433)

First, suppose that we only have noise on center of a sample. For example we received D = $[9\pm\alpha;\ 0;\ 2;\ 2]$ instead of D = [9; 0; 2; 2]. We want to find the maximum value of α that doesn't alter our clustering. After executing this experiment, α =2.09.

For example if D = [6.91; 0; 2; 2], we have the following dendrogram. This dendrogram shows the maximum noise on the center of sample D that can be supported. If the noise is bigger than 2.09 then we have different clusters.

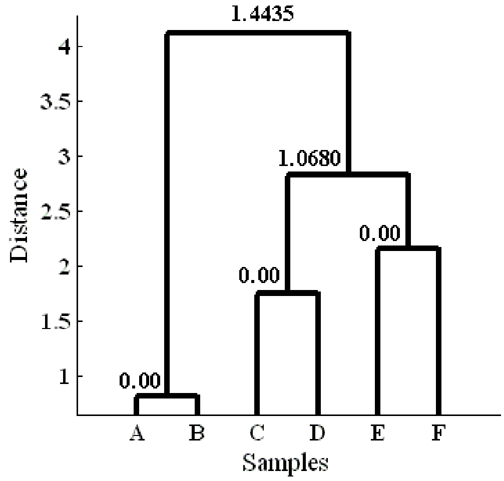

Fig. 9. Fuzzy dendrogram for Example 3.
(After adding noise to center of D)

So, if we suppose threshold=1.4433 and run ET algorithm on our samples, it can tolerate %23.2 noise on center of D. Table 6 shows the rest of the results.

Table 6: Percent of noise tolerance for Fig. 7 when we have noise on center of samples.

Samples	A	B	C	D	E	F
The value of α	1.85	1.76	1.02	2.09	3.16	1.61
The percent of noise tolerance for center of fuzzy samples	46	44	12.7	23.2	31.6	14.6

We can do this experiment on right, left and length of a fuzzy sample instead of its center.
As it can be seen, we presented a fuzzy data by four crisp data, and that is possible to have noise on any of them.

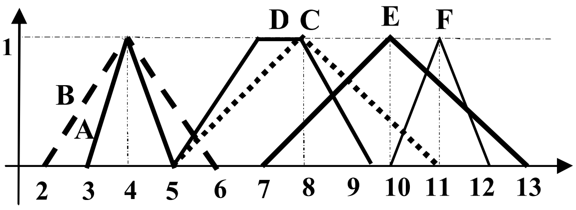

Fig. 10. Representation of Fig. 7 samples with noise on center, length, left and right of D

Now, suppose that we have noise on center, length, left and right of a sample simultaneously (see Fig. 10). For example we received D = $[9\pm\alpha;\ 0\pm\beta;\ 2\pm\lambda;\ 2\pm\gamma]$ instead of D = [9; 0; 2; 2]. We can't find the maximum value of α, β, λ and γ that do not alter our clustering, because they are mutually depend on each other. But we can consider results in a special state when we suppose that α=β=λ=γ. In this way we can find out the minimum noise that can be tolerated by this algorithm. For example, if according to Fig. 10, we received D= [7.5; 1; 2.5; 2] instead of D= [9; 0; 2; 2] we have the following results.

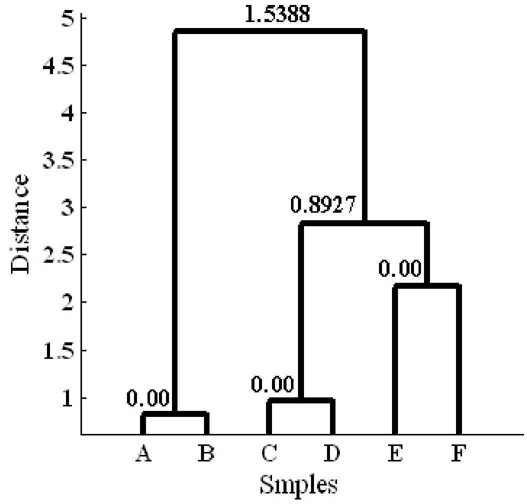

Fig. 11. Fuzzy dendrogram after adding noise on center, length, left and right of D.

So, we have the same two previous clusters (C1= {A, B}, C2={C, D, E, F}) and our clustering remained without any change. In Table 7, you can see the minimum noise on fuzzy data (used in Example 3) that can be tolerated. These noises can be added to each part of a fuzzy sample (center, length, left and right) without change of clustering.

Table 7: The minimum noise that can be tolerated for each samples in Fig. 7.

Samples	A	B	C	D	E	F
The value of noise (α)	1.20	1.06	0.65	1.34	1.40	0.91
The percent of noise tolerance for center of fuzzy samples	30	26.5	8.1	14.9	14	8.3

In Table 7 we suppose that a fuzzy data is shown as $[a_1 \pm\alpha;\ a_2 \pm\alpha;\ a_3 \pm\alpha;\ a_4 \pm\alpha]$, that α is the minimum noise which can be tolerated by ET algorithm. In this table, we just compute the percent of noise tolerance for center of fuzzy samples, because center of fuzzy samples are very important to cluster data. We can compute the percent of noise tolerance for other part of fuzzy samples (length, left and right) similarly.
According to Table 7, we can tolerate noise on inner samples of clusters more than boundary samples. Finally consider that the value of noise tolerated is different in any example and based on the distance of clusters from each other.
Example: We consider a data set G with 20 triangular fuzzy data [11]. Intuitively, the number of clusters that is suitable for data set G is two (Fig. 12).

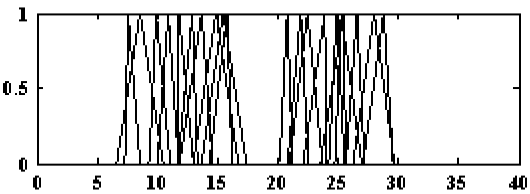

Fig. 12. Data set G with 20 triangular fuzzy data.

If we run our algorithm on these fuzzy numbers, a dendrogram is formed and we can easily extract the clusters from that, by removing the highest link of dendrogram or via inconsistency coefficient measure by selecting threshold between 1.5465 and 2.9078 (Here we use the mean of them so threshold equal to $(1.5465+2.9078)/2 = 2.22715$). In this state ET, FCM and AFCN gave the same results.
Now, we repeat these algorithms on this data set after adding a noisy sample to it. We add a point (100; 0.71; 1.79) to the data set G. The added point is far away from the other TFNs so that it can be regarded as an outlier. In this case we reach the following dendrogram.

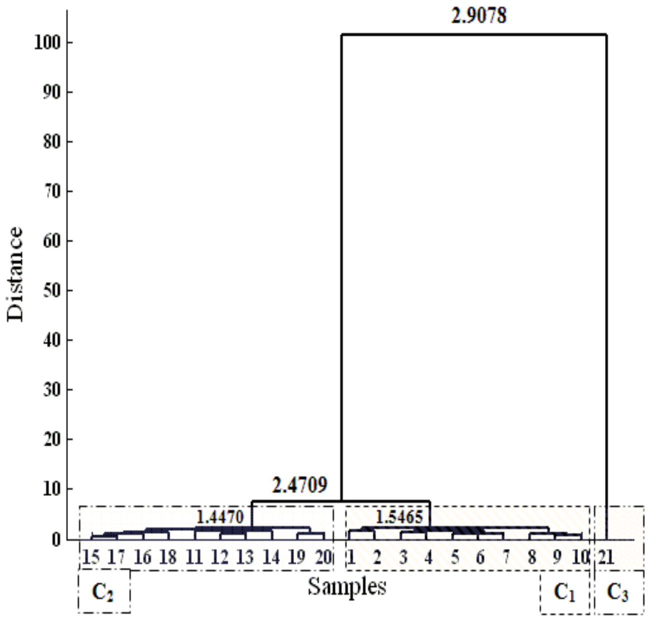

Fig. 13. Fuzzy dendrogram for data set G with a noisy sample

Table 8 shows the clustering results of ET, FCM and AFCN after adding a noisy sample. So we have three clusters C_1, C_2 and C_3 that are distinguished in Fig. 13, because the inconsistency coefficient of two levels of dendrogram are higher than threshold (Note that threshold is 2.22715). Table 8 shows that FCM cannot tolerate noisy samples, AFCN is better than FCM but ET is the best because the noisy samples are classified in the separate clusters and the noisy samples cannot change the other clusters.

Table 8: Clustering results in data set g with a noisy sample using FCM, AFCN, ET.

No	TFNs	FCM	AFCN	ET
1	(7.56, 0.27, 1.00)	C_1	C_1	C_1
2	(8.56, 1.95, 1.93)	C_1	C_1	C_1
3	(9.89, 0.56, 1.17)	C_1	C_1	C_1
4	(10.89, 0.89, 0.88)	C_1	C_1	C_1
5	(11.78, 0.12, 1.21)	C_1	C_1	C_1
6	(12.90, 1.19, 0.41)	C_1	C_1	C_1
7	(13.67, 1.82, 0.90)	C_1	C_1	C_1
8	(14.87, 1.90, 1.85)	C_1	C_1	C_1
9	(15.45, 1.79, 1.95)	C_1	C_1	C_1
10	(15.78, 1.47, 0.42)	C_1	C_1	C_1
11	(20.77, 0.63, 0.47)	C_1	C_2	C_2
12	(21.88, 1.08, 0.66)	C_1	C_2	C_2
13	(22.45, 1.48, 1.26)	C_1	C_2	C_2
14	(23.88, 1.79, 0.16)	C_1	C_2	C_2
15	(24.88, 0.66, 0.64)	C_1	C_2	C_2
16	(25.25, 0.52, 1.71)	C_1	C_2	C_2
17	(25.47, 1.95, 0.15)	C_1	C_2	C_2
18	(26.56, 0.92, 0.63)	C_1	C_2	C_2
19	(27.98, 1.74, 1.69)	C_1	C_2	C_2
20	(28.77, 1.71, 0.79)	C_1	C_2	C_2
21	(100.00, 0.71, 1.79)	C_2	?	C_3

6 Conclusions

In this paper we described a new approach for clustering fuzzy data. So far a few papers have presented methods to cluster fuzzy data that are based on fuzzy c-means algorithm. Here we open a new point of view to cluster fuzzy data based on hierarchical clustering methods. In this approach computing the distance between fuzzy data and drawing fuzzy dendrogram, lead to forming clusters. The experimental results demonstrated the major advantage of ET in comparison with similar methods that is fault tolerant against noisy samples. Furthermore the fuzzy dendrogram can present us a general view of the relation between the fuzzy data that help us to cluster them more accurately. ET is a very suitable clustering algorithm for fuzzy samples. The nature of this method is illustrative and clear so that we can guesstimate dispersion of clusters, with a glance at dendrogram. A future work might be presenting ET based on Median algorithm and building a new classifier based on this method.

7 Acknowledgments

This work has been partially supported by Iran Telecommunication Research Center (ITRC), Tehran, Iran. This support is gratefully acknowledged.

8 References

[1] A. R. Webb, “Statistical Pattern Recognition”, Wiley, New York, 2002.

[2] R. O. Duda, P. E. Hart, D. G. Stork. “Pattern classification”, Wiley, New York, 1997.

[3] J. C. Bezdek, “Pattern recognition with fuzzy objective function algorithms”, Plenum press, New York, 1981.

[4] M. S. Yang, P. Y. Hwang, D. H. Chen, “Fuzzy clustering algorithms for mixed feature variables”. Fuzzy Sets and Systems 141, pp. 301–317, 2004.

[5] S. M. Chen, L. Y. Chen, “A Fuzzy hierarchical clustering method for clustering documents based on dynamic cluster centers”, Journal of the Chinese Institute of Engineers, Vol. 30, No. 1, pp. 169-172, 2007.

[6] P. Grzegorzewski, “Metrics and orders in space of fuzzy numbers”, Fuzzy sets and systems 97, pp. 83-94, 1998.

[7] V. Torra, “Fuzzy c-means for fuzzy hierarchical clustering”, IEEE transactions on Fuzzy Systems, Vol. 14, pp. 646-651, 2005.

[8] A. B. Geva, “Hierarchical Unsupervised Fuzzy Clustering”, IEEE transactions on fuzzy systems, Vol. 7, No. 6, pp. 723-733, 1999.

[9] R. O. Duda, P. E. Hart, “Pattern classification and scene Analysis”, Wiley, New York, 1973.

[10] L. Stefaninia, L. Sorinia, M. L. Guerraa, “Parametric representation of fuzzy numbers and application to fuzzy calculus” Fuzzy Sets and Systems 157, pp. 2423-2455, 2006.

[11] W. L. Hung, M. S. Yang, "Fuzzy clustering on LR-type fuzzy numbers with an application in Taiwanese tea evaluation", Fuzzy Sets and Systems 150, pp. 561–577, 2005.

[12] R. J. Hathaway, J. C. Bezdek, W. Pedrycz, “A parametric model for fusing heterogeneous fuzzy data”, IEEE transactions on Fuzzy Systems, Vol. 4, No. 3, 1996.

[13] P. C. Wang, J. J. Leou, “New Fuzzy Hierarchical Clustering Algorithms”, Journal of Information Science and Engineering, Vol. 9, No. 3, pp. 461-489, 1993.

[14] M. Sato, Y. Sato, “Fuzzy clustering model for fuzzy data”, IEEE transactions on Fuzzy Systems, Vol. 4, pp. 2123 – 2128, 1995.

[15] M. S. Yang, C. H. Ko, “On a class of fuzzy c-numbers clustering procedures for fuzzy data”, Fuzzy Sets and Systems 84, pp. 49–60, 1996.

[16] Available: http: //www.mathworks.com