CHAPTER III

GENETIC-FUZZY-CLUSTERING ALGORITHMS FOR IMAGE COMPRESSION
Chapter III

GENETIC-FUZZY-CLUSTERING ALGORITHMS FOR IMAGE COMPRESSION

3.1. TECHNIQUE FOR FRACTAL IMAGE COMPRESSION USING GENETIC ALGORITHM

The theory of image coding using iterative function system (IFS) was first proposed by Barnsley [11]. He modeled real life images by means of deterministic fractal objects, that is, by the attractors evolved through iterations of a set of contractive affine transformations. With the help of iterated function system, along with college theorem, Barnsley laid the foundation of the fractal-based image compression. A set of contractive affine transformation-iterative function system (IFS)-can approximate a real image and so, instead of storing the whole image, it is enough to store the relevant parameters of the transformations reducing the requirement of memory. The basic problem of fractal-based image compression is to find appropriate parameter values of transformation whose attractor is an approximation of the given image. A review on fractal-based image coding methodology is available in the literature [96].

A fully automated fractal-based image compression technique of digital monochrome image was first proposed by Jacquin [95]. The encoding process consists of approximating the small image blocks, called range blocks, from the larger blocks, called domain blocks, of the image, through some operations. In the encoding process, separate transformation for each range block is obtained. The scheme also uses the theory of vector quantization [175] to classify the blocks. The set consists of these transformations, when iterated upon any initial image, will produce a fixed point (attractor) that approximates the target images. This scheme can be viewed as partitioned iterative function system (PIFS). One such scheme, using PIFS, to store fewer numbers of bits (or to increase the compression ratio) was proposed by Fisher et al. [60].
Genetic algorithms (GA's) [21,41,61,66,160,161] are mathematically motivated search techniques that try to emulate biological evolutionary processes to solve optimization problems. Instead of searching one point at a time, GA's use multiple search points. GA's attempt to find near-optimal solutions without going through an exhaustive search.

A new method for image compression using PIFS is proposed. The proposed method uses a simpler classification system for range blocks. Genetic algorithms with elitist model are used in finding the appropriate domain block as well as the appropriate transformation for each range block. An analytical study of the proposed method along with a comparison with other existing methods is also reported here.

**Theory and basic principles**

(i) **Theoretical foundation of image coding by iterative function system**

The details mathematical descriptions of the IFS theory, collage theorem, and other relevant results are available [11,53,58]. Only the salient features of image coding through IFS are given below.

Let I be a given image that belongs to the set X. Generally, X is taken as the collection of compact sets. Our intention is to find a set \( \mathcal{F} \) of affine contractive maps for which the given image I is an approximate fixed point \( \mathcal{F} \) is constructed in such a way that the distance between the given image and the fixed point (attractor) of \( \mathcal{F} \) is very small. The attractor "A" of the set of maps \( \mathcal{F} \) is defined as follows

\[
\lim_{N \to \infty} \mathcal{F}^N(J) = A \quad \forall J \in X \quad \ldots (3.1.1)
\]

and \( \mathcal{F}(A) = A \), where \( \mathcal{F}^N(J) \) is defined as

\[
\mathcal{F}^N(J) = \mathcal{F}[\mathcal{F}^{N-1}(J)],
\]

with

\[
\mathcal{F}^1(J) = \mathcal{F}(J), \forall J \in X
\]

Also, the set of maps \( \mathcal{F} \) is defined as follows
\[ d[F(J_1), F(J_2)] \leq s \, d(J_1, J_2); \quad \forall J_1, J_2 \in X \text{ and } 0 \leq s < 1. \quad \text{(3.1.2)} \]

Here, \( d \) is called the distance measure and \( s \) is called the contractive factor of \( F \): Let

\[ d[I, F(I)] \leq \varepsilon \]

where \( \varepsilon \) is a small positive quantity. Now, by the college theorem [11], it can be shown that

\[ d(I, A) \leq \frac{\varepsilon}{1 - s} \]

where \( A \) is the attractor of \( F \).

From (3.1.3) it is clear that, after a sufficiently large number \((N)\) of iterations, the set of affine contractive maps \( F \) produces a set that belongs to \( X \) and is very close to the given original image \( I \). Here, \((X, F)\) is called iterative function system and \( F \) is called the set of fractal codes for the given image \( I \).

**(ii) Basic Principles and Features of Genetic Algorithms**

GA's are adaptive search process based on the notion of select mechanism of natural genetic system [66]. GA's help to find the global near optimal solution without getting stuck at local optimal as they deal with multiple points (spread all over the search space) simultaneously. To solve the optimization problem, the GA starts with the structural representation of a parameter set. The parameter set is coded in a string of finite length and the string is called chromosome. Usually, the chromosomes are strings of zero's and one's. If the length of string (chromosome) is 1, then the total number of possible strings is 2.

**Fitness Function:** Usually, a function, called fitness function is defined on the set of chromosomes (strings). The problem here is to find the string (chromosome) that provides optimal fitness value among all strings (chromosomes). In this work, we are dealing with a minimization problem.
Initial Population: Out of all possible $2^t$ strings, initially a few strings (say $S$ number of strings) are selected randomly and this set of strings is called initial population [66]. Here, we have taken $S$ to be an even numbers.

Three basic genetic operators, i) Selection, ii) crossover, and iii) mutation, are exploited in GA's. The genetic operators are applied on the initial population rise to a new population of same size ($S$). The operators are again applied on this new population to give rise to another population. The process of creation is executed for a fixed number of iterations. It is proved that the elitist model of GA will find the optimal solution as the number of iterations tends to infinity. Usually, for real-life problems, stopping times of the GA are satisfactorily found after several experiments. For the present problem too, the stopping time is found after several experiments. The descriptions of the different operations are given below.

Selection: In this operation, a mating pool of strings of the current population is generated by using the fitness value of strings in the population. The probability of selection of a string in the population to the mating pool is inversely proportioned to its fitness value. (Note that the present optimization problem is a minimization problem). This scheme is known as proportional selection having optimal fitness value among all the strings in the present population. The size or the number of strings in a matting pool is the same as it is of the initial population.

Crossover: There are several ways of performing the crossover operation [41] among the strings in the mating pool. The single point crossover operation is followed as here. The crossover probability is represented by $P_{\text{cross}}$.

In this operation, $S / 2$ pairs of strings are formed randomly from the mating poll, of size $S$. A random number $r$ and in the range $[0, 1]$ is then generated for each part of strings. If $r$ and $\leq P_{\text{cross}}$, then crossover is performed. Otherwise, the operation is not performed. Each pair of strings undergoes
crossing over in the following manner. An integer position \( k \) is selected randomly between \( l \) and \( l-1 \) (\( l \geq 2 \), is the string length). Two new strings are created by swapping all the characters from position \( k + 1 \) to \( l \) of the old strings. Usually a high value is assigned for the crossover probability \( P_{\text{cross}} \)

**Mutation:** In this operation bit of ever string is flipped (i.e., zero by one and one by zero) with probability \( P_{\text{mut}} \). \( P_{\text{mut}} \) is called the mutation probability.

One of the commonly used conventions is to assign a very small value to the mutation probability \( P_{\text{mut}} \) and keep that \( P_{\text{mut}} \) fixed for all the iterations, i.e., the value for \( P_{\text{mut}} \) is independent of the number of iterations. A different strategy is adopted here in assigning the value for \( P_{\text{mut}} \). We have prefixed the number of iterations of GA a priori and varying mutation probability with the number of iterations. This varying mutation probability scheme has already been applied successfully in connection with an application of GA’s to pattern recognition problem [66].

**Elitist Model:** In the elitist model [138] of GA’s, the knowledge about the best string obtained so far is usually preserved within the population. For this purpose, the worst string in the present population is replaced by the best string of the previous population in each iteration. The best strings of two consecutive iterations will be always in the mating pool of the present population in elitist model. Thus, this model will provide a track of best strings in all the iterations.

**Proposed methodology**

(i) **Construction of Fractal Codes**

Let, \( I \) be a given image having size \( w \times w \) and the range of gray level values be \([0, g]\). Thus, the given image \( I \) is a subset in \( \mathbb{R}^3 \). The image is partitioned into non-overlapping squares of size, say \( b \times b \), and let this partition be represented by
$R = \{R_1, R_2, \ldots, R_n\}$. Each $R_i$ is named as range block. Note that $n = w/b \times w/b$. Let $D$ be the collection of all possible blocks which is of size $2b \times 2b$ and let $D = \{D_1, D_2, \ldots, D_m\}$. Each $D_j$ is named as domain block with $m = (w-2b) \times (w-2b)$.

Let $F_j = \{f: D_j \rightarrow \mathbb{R}^3; f$ is an affine contractive map$\}$. Now, for a given range block $R_i$, let $f_{ij} \in F_j$ be such that

$$d[R_i, f_{ij}(D_j)] \leq d[R_i, f(D_j)] \ \forall f \in F_i, \forall j.$$  

Now let $k$ be such that

$$d[R_i, f_{ij}(D_k)] = \min \{d[R_i, f(D_j)]\}. \quad \ldots(3.1.4)$$

Also, let $f_{ik}(D_k) = R_{ik}$.

Our aim is to find $f_{ik}(D_k)$ for each $i \in \{1, 2, \ldots, n\}$. The set of maps $F = \{f_{1\ast}, f_{2\ast}, \ldots, f_{n\ast}\}$ thus obtained is called the fractal code of image I. Also note that there are other works on fractal image compression in which the domain and range blocks are utilized [60, 95].

The find the best matched domain block as well as the best matched map, one has to search all possible domain blocks with the help of (3.1.4). The affine contractive map $f_{ik\ast}$ is constructed in two steps. The first step is transformation of rows and columns from domain blocks to range blocks. This part is nothing but the change of coordinates in a two-dimensional (2-D) geometry. This can be achieved by using any one of the eight possible transformations, called isometrics, one the domain blocks [95]. Once the first part is obtained, second part is estimation of a set of pixel values of range blocks from the set of pixel values of the transformed domain blocks. These estimates can be obtained by using the least square analysis of two sets of pixel values.

The distance measure "d" [used in (3.1.4)] is taken to be the simple mean square error (MSE) between the original set of gray values and the
obtained set of gray values and the obtained set of gray values of the concerned range block, viz., \( R_i(p, q) \) and \( R_{ih}(p, q) \), respectively. As selection of fractal code for a range block is dependent only on the estimation of pixel values of that block, it is enough to calculate only the distortion of the original and estimated and estimated pixel values of the block. Thus, the MSE is taken as the distance measure. Note that the same measure had also been used in other works [60, 95]. The fractal code \( \mathcal{F} \) is used for decoding the given image \( I \) from any arbitrary starting image \( I_0 \).

A two-level image partition scheme as reported by Jacquin [95] is used for the implementation of the GA-based fractal image compression scheme.

The GA is used to search for an appropriately matched domain block as well as an appropriate transformation for a particular type of range block. The class of range blocks is obtained through a simple classification scheme.

(ii) Classification

The purpose of block classification is twofold. One purpose is to store fewer numbers of bits or to get higher compression ratio and the other is to reduce the encoding time. A simple classification scheme on range blocks alone is used here. Range blocks are grouped into two sets according to the variability of the pixel values in these blocks. If the variability of a block is low, i.e., if the variance of the pixel values in the block is below a fixed value, called the threshold, we call it a rough type range block. The threshold value that separates the range blocks into two types is obtained from the valley in the histogram of the variances of pixel values within blocks. After classification, GA-based encoding is adopted for rough type range blocks. All the pixel values in a smooth type range block are replaced by the mean of its pixel values. The scheme mentioned above is a time saving one provided the number of smooth type range blocks is significant. The analysis of the proposed method discusses these aspects.
(iii) Genetic Algorithm to find Fractal Codes

The main aspect of fractal-based image coding is to find a suitable domain block and a transformation for a rough type range block. Thus, the whole problem can be looked upon as a search problem. Instead of a global search mechanism we have introduced GA's to find the near optimal solution.

The number of possible domain blocks to be searched are \((w - 2b) \times (w - 2b)\) and the number of transformations to be searched for each domain block is eight. Thus, the space to be searched consists of \(M\) elements. \(M\) is called cardinality of the search space. Here, \(M = 8(w - 2b)^2\). Let the space to be searched be represented by \(\mathcal{P}\) where

\[
\mathcal{P} = \{1, 2, \ldots, (w - 2b)\} \times \{1, 2, \ldots, (w - 2b)\} \times \{1, 2, \ldots, 8\}.
\]

Binary strings are introduced to represent the elements of \(\mathcal{P}\). The set of \(2^l\) binary strings, each of length \(l\), are constructed in such a way that the set exhausts the whole parametric space. The value for \(l\) depends on the values of \(w\) and \(b\). The fitness value of a string is taken to be the MSE between the given range block and the obtained range block.

Let \(S\) be the population size and \(T\) be the maximum number of interactions for the GA. Note that the total number of strings searched up to \(T\) iterations is \(ST\). Hence, \(M / ST\) provides the search space reduction ratio for each rough type range block.

**Implementation and results**

The GA based method discussed as early is implemented in \(256 \times 256\), 8 b/pixel Lena image. The image is subdivided into four, \(128 \times 128\) sub images, each of which is encoded separately.
Table 3.1.1: Test results for $256 \times 256$, 8 b/pixel lena image

<table>
<thead>
<tr>
<th>Type of encoding</th>
<th>Block Size</th>
<th>Number</th>
<th>Compression ratio</th>
<th>Bits per pixel</th>
<th>PSNR (in db)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Range</td>
<td>Domain</td>
<td>Parent</td>
<td>Child</td>
<td></td>
</tr>
<tr>
<td>Single level</td>
<td>$8 \times 8$</td>
<td>$16 \times 16$</td>
<td>278</td>
<td>746</td>
<td>Nil</td>
</tr>
<tr>
<td></td>
<td>$8 \times 8$ and $4 \times 4$</td>
<td>$16 \times 16$ and $8 \times 8$</td>
<td>278</td>
<td>746</td>
<td>Nil</td>
</tr>
</tbody>
</table>

For the specific implementation of the classification scheme mentioned above, the variances of pixel values of all $8 \times 8$ and $4 \times 4$ range blocks are computed and corresponding thresholds are selected from the respective histograms of the variances. For $8 \times 8$ range blocks, a valley is found near 20 in the histogram and thus, this value is chosen as threshold for $8 \times 8$ range blocks. Similarly 35 is taken as the threshold value for the $4 \times 4$ range blocks.

Considering parent range blocks of size $8 \times 8$ and children range blocks of size $4 \times 4$ and using two level image partition scheme [95] each sub image is then encoded. The methodology proposed here is also implemented with a single level partition scheme when only parent blocks are considered.

GA’s are implemented, as a search technique, only for rough type range blocks. Here for each sub image, total number of parent range blocks is $n = 256$ and total number of domain blocks($m$) to search is $(128 - 16) \times (128 - 16) = 112 \times 112$ and $(128 - 8) \times (128 - 8) = 120 \times 120$ for parent and child range blocks, respectively. Thus, the cardinality ($M$) of the search spaces for these two cases are $112 \times 112 \times 8$ and $120 \times 120 \times 8$, respectively. The string length 1 has been taken to be $17(7 + 7 + 3)$ in both the cases. A few strings, in both the cases will be outside the specified search spaces. If a string which is not in the search space selected during the implementation of the GA, then a string at the boundary will replace it.
Out of these $2^{17}$ binary strings, six strings ($S = 6$) are selected randomly to construct an initial position. A high probability, say $P_{\text{cross}} = 0.85$, is taken for the crossover operation. For mutation and the exact values are 0.30, 0.20, 0.15, 0.10, and 0.06. The total number of iterations considered in the GA is $T = 910$. Hence, the search space reduction of the child rough type range blocks are approximately 18 and 21 considered respectively.

For encoding the Lena image, the results of both one level (i.e only $8 \times 8$ range blocks) and two level (i.e first $8 \times 8$ and the $4 \times 4$) encoding are reported here. Test results and some statistics of both the cases are given in table 3.1.1.

The GA based technique of fractal image compression method is compared with exhaustive search mechanism. The test results of single level and the two level encoding schemes of both the techniques are shown in table 3.1.2. Note that all the prefixed parameters are the same in both cases.

It is clear from the test results that for single level encoding the compression ratios are found to be same in both techniques. On the other hand, the Peak-signal noise ratio (PSNR) values are very close to each other. But the number of domain blocks searched in both cases provides the justification of using the GA as a search technique. The number of domain blocks searched in cases of GA is at least 20 times smaller than that of the case of exhaustive search technique. These blocks are not divided into child blocks for second level encoding. As a result of this, more compression is achieved in exhaustive search case. But the PSNR value appears to be better in case of GA – based technique. Moreover, the advantage of using the GA – based technique is established from the value of the number of domain blocks searched in both the cases. Here also, the GA – based techniques searched at least 20 times fewer domain blocks in comparison with the exhaustive search techniques.

The search space reduction in achieved, since near – optimal solutions are usually satisfactory, and intuitively, the solution whose fitness values are far
away from the optimal are thrown away in a bulk. This is the reason GA performs well for optimization problems [41].

Comparison and analysis

(i) Comparison

Advantages of GA - based fractal image coding are clearly demonstrated in reducing the search space for finding the appropriate matched domain block and the appropriate transformation corresponding to a range block. The GA method is found providing computational efficiency, thereby drastically reducing the provided computational efficiency, thereby drastically reducing the cost of coding. The obtained results for the Lena image are also comparable with that of some of the existing methods of fractal image compression [60, 95].

<table>
<thead>
<tr>
<th>Type of encoding</th>
<th>Genetic Algorithm</th>
<th>Exhaustive Search</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Compression Ratio</td>
<td>PSNR in db</td>
</tr>
<tr>
<td>Single level 8 x 8</td>
<td>21.7</td>
<td>26.16</td>
</tr>
<tr>
<td>Two level 8 x 8 and 4 x 4</td>
<td>10.50</td>
<td>30.20</td>
</tr>
</tbody>
</table>

A different kind of trimming of the search space was described by Jacquin [95]. In his method reduction takes place in two steps, by making a “domain pool” consisting of domain blocks and then by “classifying” this pool in some classes based on the geometric features of the blocks.
In Jacquin's method starting from the first pixel of the given image, domain blocks are selected by sliding a window of size equal to the size of the domain blocks across the image and taking a constant shift horizontally and vertically. Shift of four pixels and two pixels are considered for selecting the domain pool consisting of domain blocks of size $16 \times 16$ and $8 \times 8$ respectively. Thus the reduction ratios are 16 and 4 for $8 \times 8$ and $4 \times 4$ range blocks, respectively, for a $256 \times 256$ image. On the other hand, the GA-based method reduces the search space corresponding to domain blocks and isometric transformations simultaneously, and the search space production ratios for a parent and a child rough type range blocks are approximately 18 and 21 respectively. Moreover, the best matched domain block corresponding to a range block can be located anywhere within the image support, and so, on trimming the maximal domain pool by shift method, we may lose the best matched domain block in the Jacquin's method. But the GA-based method utilizes the maximal domain pool while searching for the best matched domain block.

The second part of the reduction is obtained using the classification scheme proposed by Ramamurthy and Grasho [175]. This three class classification scheme is adopted to classify both the pool of range blocks as well as domain blocks. This scheme has advantages both in efficient encoding of the range blocks as well as in reducing its search space. But it has limitations, too. According to the scheme, both pools are classified into three classes, so 2b may be required for storing to indicate the class of the range block under consideration. In the present encoding algorithm, only the range blocks are classified into two classes, which is not only a time-saving scheme, but it also requires storing of only 1b for class information thereby providing more compression.

GA-based fractal image compression scheme discussed here is also comparable with the algorithm given by Fisher et al. [60]. There is a transformation for each range block and the parameter $s$ of transformations is
stored instead of whole image. Thus, the compression ratio depends on the number of range blocks. More compression can be achieved by considering less number of range blocks. Considering fewer number of range blocks by using quad tree method and $h - v$ partitioning method [60], they designed their scheme, which results in the increase of compression ratio. The compression ratio (9.97) and the PSNR (31.53) reported by Fisher et al. [60] are almost equal to those reported here (Table 3.1.1).

**Table 3.1.3: Number of bits to be stored in different schemes of a 256×256, 8 b/pixel image during the decoding process**

<table>
<thead>
<tr>
<th></th>
<th>Single level (8 × 8)</th>
<th>Two level (8 × 8) and (4 × 4)</th>
<th>Mixed (Parent and child)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>All parent</td>
<td>All child</td>
</tr>
<tr>
<td><strong>With Classification</strong></td>
<td>28 × 1024</td>
<td>28 × 1024 + 1024 × 4</td>
<td>28 × 4096 + 1024 × 4</td>
</tr>
<tr>
<td><strong>All smooth</strong></td>
<td>9 × 1024</td>
<td>9 × 1024 + 1024 × 4</td>
<td>9 × 4096 + 1024 × 4</td>
</tr>
<tr>
<td><strong>All rough</strong></td>
<td>29 × 1024</td>
<td>29 × 1024 + 1024 × 4</td>
<td>29 × 4096 + 1024 × 4</td>
</tr>
<tr>
<td><strong>Mixed (smooth and rough)</strong></td>
<td>29 × 1024 – 20 × r</td>
<td>29 × 1024 – 20 × r + 1024 × 4</td>
<td>29 × 4096 – 20 × r + 1024 × 4</td>
</tr>
</tbody>
</table>

The search space in the methodology described by Fisher et al. [60] is dependent on the complexity of the given image. In particular, in the first step of quadtree method, best domain block for only four range blocks has to be searched. The search process is then carried out up to a fixed level where the minimum size of the range blocks is fixed. The search, in all the intermediate steps, has to be done exhaustively to reach the fixed lowest level in the quadtree method. Thus, an extensive search may need to be carried out for some images. So, in comparison with this method, the proposed GA-based method is better for reducing the search space. Note that, the proposed GA-based can be extended to
a quadtree scheme containing multiple levels, with or without classification of range blocks.

(ii) Analysis

A two level image partition scheme (parent and its four children) as implemented takes care of finer details of a very small portion of the image. There are altogether 12 possible configurations of four children along with their parents. To indicate the location and presence of child blocks of a parent block, four extra bits for each parent block are needed during encoding. Likewise, an extra bit for each transformation (parent and child) is needed if the classification scheme together with the two level image partition scheme need to be investigated. Comparison of performances of different encoding schemes (single level, two level, without classification, with classification) can be made from the point of view of compression and quality of the decoded image. Table 3.1.3 shows the number of bits required for a $256 \times 256$, 8 b/pixel image under different situations in different schemes. Number of range blocks are 1024 and 4096 with size $8 \times 8$ and $4 \times 4$, respectively. Number of bits require to store each transformation is 28 in the coding scheme not using classification scheme. Coding scheme using classification requires store 9b and 29b respectively, for the transformation of each smooth type and rough type range blocks.

Table 3.1.3 shows the number of bits to be stored with different schemes for a $256 \times 256$, 8 b/pixel image. The selection of a particular scheme for a given image, which provides suitable values for the PSNR and the compression ratio, depends upon the values of “c” and “r”. Here, “c” is the total number of codes in two level partitioning where both parent and child blocks are present and “r” is the number of smooth type range blocks under classification scheme. The following conclusions regarding the usefulness of the application of different schemes during encoding process may be extracted from table 3.1.3.
If a "single level" scheme, provides the desired quality of the decoded image (ie., the desired PSNR), then one should not opt for a "two level" scheme. Note that a two level scheme decreases the compression ratio.

If all the blocks in a "two level" scheme are of same size, then "single level" encoding with that block size is preferable.

Under no classification, a "two level" encoding scheme where both parent and child blocks are present is preferable to "single level" scheme if \( c < 877 \). On the other hand, "single level" encoding with block size 4 \( \times \) 4 is preferable to this scheme when \( c > 3950 \). If \( c \) lies between 877 and 3950, then switching from "single level" to "two level" depends upon the quality of the decoding image.

Under classification, when all the blocks are "smooth" type, a "two level" encoding scheme where both parent and child blocks are preferable to "single level" encoding with block size 4 \( \times \) 4 is preferable to this scheme when \( c > 3641 \). If \( c \) lies between 568 and 3641, then switching over from "single level" to "two level" depends upon the quality of the decoded image.

Similarly, with classification, when all the blocks are "rough" type or "mixed" type (both smooth and rough type blocks are present), a "two level" encoding scheme where both parent and child blocks are preferable to "single level" scheme if \( c < 880 \). On the other hand, "single level" encoding with block size 4 \( \times \) 4 is preferable to this scheme when \( c > 3955 \). If \( c \) lies between 880 and 3955 then switching over from "single level" to "two level" depends upon the quality of the decoded image.

Let \( r_1 \) be the number of parent smooth blocks out of \( c \) number of codes. Let \( 29 \times c + 60 \times r_1 > 28 \times 4096 \) and also let every child block and parent block be a smooth block. (Note that there are four 4 \( \times \) 4 child blocks of each 8 \( \times \) 8 parent block). Then, "single level with classification" scheme with block size 4 \( \times \) 4 is preferable to "two level with classification" scheme.

If all the blocks belong to the same class then "classification" is not needed.
If both smooth and rough type blocks are present with number of smooth blocks less than 5% of the total codes present then classification is not needed.

Note that similar inferences may be drawn for the other available fractal-based image coding schemes.

The effectiveness of the GA-based fractal image compression technique depends upon three factors: i) the number of points in the search space $2^l$, ii) the size of the initial population $S$, and iii) the number of iterations $T$. The number of iterations will be different for different images to achieve the near-optimal solution using GA’s.

PSNR is used here to measure the quality of the decoded image. There are other measures for judging the quality of the decoded image. It is to modify the proposed method for other measures of judging the quality of the decoded image. MSE is used in the fitness function for the set of strings are presented. Any other suitable function, which measures the distortion between the given range block and the obtained range block, can be used as the fitness function instead of MSE in the proposed method. One can achieve a high compression ratio by sacrificing the quality of the decoded image. On the contrary, a high-quality decoded image can be obtained at the cost of compression ratio. Thus, a trade-off has to be made to obtain a good quality decoded image with a considerable amount of compression. A stopping criterion of the decoding process in the present methodology can be suggested by using a threshold value on the difference between the resulting images of two successive iterations.

The GA-based method reported here and the quadtree-based method [60] can be applied simultaneously to give rise to a new method of fractal coding. Quadtree can be applied during the partitioning of the given image and GA’s can be applied as the search technique in each step of quadtree partitioning.
The threshold value for the classification of range blocks corresponds to the valley in the histogram of the variances of the pixel values of blocks. Note that it can not always be assured that the said histogram is strictly bimodal. The suitable values for the valleys in the histograms are obtained here by visual inspection for the images considered here. One can use any one of the several thresholding methodologies for finding valleys in the histograms.

A simple technique for the classification of the range blocks is used here. However, one can use other techniques in this regard. A technique that utilizes the psychovisual properties of human visual system has been developed recently for the classification of the image blocks.

The basic philosophy of the proposed GA-based technique can also be adopted for other fractal-based coding techniques. Thomas et al suggested an algorithm for fractal-based image compression in which the neighbourhood information plays an important role in increasing the compression ratio. In that method, the domain block for a “seed” range block is found by an exhaustive search mechanism. The domain blocks for the other range blocks are found by utilizing the connectivity of the range blocks. One can adopt the proposed GA based technique for finding the domain block for the “seed” range block.

3.2. A FUZZY ALGORITHM FOR COLOUR QUANTIZATION OF IMAGES

Most colour image printing and display devices do not have the capability of reproducing true colour images, which may contain upto 16.6 million colours. Prior to display or printing of true color images, the number of colours in an image is reduced by a process called colour quantization. The reduced colours constitute a colour palette, which typically contains 256 or fewer entries.

The fastest colour quantization methods produce a colour palette by iteratively partitioning the colour space with planes perpendicular to the colour
axes. These techniques produce images with mediocre quality, however, due to their tree structured palettes, they afford fast execution. Orchard and Bouman have devised a better tree-based approach which partitions the colour space with planes perpendicular to the principal axis of the colour covariance matrix. Techniques based on clustering schemes such as the C-means algorithm produce better quality palettes at higher computational cost.

Fuzzy quantization is a generalization of hard quantization schema and the best known and most widely used fuzzy quantization technique is the fuzzy C-means (FCM) algorithm. In the FCM algorithm, each data point belongs to a cluster with a degree specified by a membership grade between zero and one. Since the colour space of the images and the colour clusters contained in this colour space are irregular in shape and density, finding representative colours for colour clusters is a suitable application area for fuzzy techniques. The FCM algorithm produces locally optimal solutions and generally produces better palettes than the hard C-means algorithm.

Recently, new fuzzy quantization techniques for joint quantization and dithering have been proposed. In combined approaches, the basic idea is introducing the dithering error to the quantizer to output an optimized codebook that is more suitable for dithering. Therefore the dithering error affects the resultant codebook the quantizer produces.

We presents a short summary of fuzzy colour quantization techniques and presents a new technique for fuzzy colour quantization. We propose a fuzzy colour quantization technique that incorporates a partition index term to more efficiently quantize images. Our partition index term is based on the cluster validity idea. Cluster validity is concerned with checking the quality of clustering results. It has been mainly used to evaluate and compare the resultant partitions from different algorithms or from the same algorithm under different parameters. Generally, cluster validity measures are used to evaluate the
groupings of data into clusters data. Less commonly, they have been used to judge the quality of the individual clusters. In [14], it is used for the re-clustering of a fixed number of clusters through a split and merge approach to obtain a better codebook. In our work, we used the cluster validity idea to establish a partition index criterion. The resulting algorithm has an improved quantization performance and a reduced computational cost compared to the fuzzy C-means algorithm. It can be coupled with any dithering technique, preferably fuzzy dithering, to obtain visually high quality images.

**Fuzzy algorithms for colour quantization**

**Fuzzy quantization**

The first attempt for fuzzy quantization has been the adaptation of the fuzzy C-means algorithm for this purpose. In the FCM algorithm, n vectors are partitioned into c fuzzy groups. Sum of membership values is equal to unity:

\[
\sum_{i=1}^{c} u_{ij} = 1, \forall j = 1, \ldots, n
\]  

...(3.2.1)

Let the pixels of a colour image be represented by 3 x 1 vectors \( \mathbf{x}_j \) for \( 1 \leq j \leq n \) where \( n \) is the total number of pixels in the image. To simplify the equations, double indices denoting the location of the pixels have been replaced by the single index \( j \). For an image of size \( M_1 \times M_2 \), the pixel at location \( (m_1, m_2) \) corresponds to the index \( j = m_1 + m_2 M_1 \). With these definitions, the objective function is defined as

\[
J = \sum_{j=1}^{n} \sum_{i=1}^{c} (u_{ij})^m d(x_j, y_i) 
\]  

...(3.2.2)

where \( m \) is the parameter of fuzziness, \( y_j \) denotes the set of quantization colours and \( d(x_j, y_i) \) is the \( L_2 \) norm. Using the Lagrange multipliers method, the minimization of the objective function results in the membership function.

\[
u_{ij} = \left[ \frac{1}{\sum_{k=1}^{c} d(x_j, y_i)/d(x_j, y_k)} \right]^{1/(m-1)}
\]  

...(3.2.3)
and the update function is
\[ y_j = \frac{\sum_{j=1}^{n} (u_{ij})^n x_j}{\sum_{j=1}^{n} (u_{ij})^n} \] ...

**Previous work on fuzzy C-mean quantization**

In [157], the FCM algorithm is modified by incorporating separation term into the objective function \( J \) to minimize the fuzzy Euclidean distance and to maximize the inter-cluster separation (ICS). The separation \( s_i \) of a fuzzy cluster \( (i) \) is defined as the sum of the distances from its cluster center \( (y_i) \) to the center of the other \((c-1)\) clusters:

\[ s_i = \sum_{t=1}^{c} \|y_i - y_t\|^2 \] ...

Then \( s_i \) is incorporated into the objective function \( J \) to settle the quantization centers to local minima such that they will be closer to the convex hull of the image colour space than their FCM converged counterparts. The examination of the resultant palettes for test image obtained from the FCM and the ICS algorithms shows that the colour space covered by the palette produced by the ICS algorithm is larger. This enlargement provides the creation of the illusion of more colours after dithering is applied.

**Partition index maximization**

Here, we propose a new extension of the FCM algorithm which minimizes an objective function incorporating a validity index. Therefore, validity index is used not to judge the resultant partitions, but to form better partitions with respect to the validity index used. In the proposed approach, we minimize an objective function including a term for partition index. This algorithm attempts to place the cluster centers such that the membership values of the pixels are maximized. The goal is to obtain better partitions faster than the FCM algorithm. Although the intended application domain is image quantization, it is applicable to any classification scheme.
Partition index is a measure of validity similar to partition coefficient, based on using \( P_j = \ldots \) as a measure of how well the \( j^{th} \) data point has been classified. The closer a pixel is to a codebook entry, the closer \( P_j \) is to one. If a pixel becomes another to all cluster centers, the value of \( P_j \) approaches \( 1/e^{m-1} \), which is the minimum value it can have. Therefore, if we aim to minimize the fuzzy Euclidean distance measure and maximize the membership values of the partitions, the objective function becomes

\[
J(u, y) = \sum_{j=1}^{n} \sum_{i=1}^{c} (u_{j})^{c} d(x_{j}, y_{i}) - \alpha \sum_{j=1}^{n} \sum_{i=1}^{c} (u_{j})^{c} \ldots (3.2.6)
\]

Under the constraint of equation (3.2.1) the parameter \( \alpha \) controls the weight of the second term and will be further examined. Using the standard technique of Lagrange multipliers, the equation to be minimized is

\[
J_{f}(u_{j}, \lambda) = \sum_{j=1}^{c} (u_{j})^{m} d(x_{j}, y_{i}) - \alpha \sum_{j=1}^{c} (u_{j})^{m} - \lambda \left( \sum_{j=1}^{c} u_{j} - 1 \right) \ldots (3.2.7)
\]

Where \( \lambda \) is the Lagrange multiplier. The minimization of \( J_{f}(u_{j}, \lambda) \) proceeds as follows:

\[
\frac{\partial J_{f}}{\partial u_{j}} = m (u_{j})^{m-1} d(x_{j}, y_{i}) - \alpha m (u_{j})^{m-1} - \lambda = 0 \ldots (3.2.8)
\]

Solving for \( u_{j} \) yields

\[
u_{j} = \left[ \frac{\lambda}{m d(x_{j}, y_{i}) - \alpha} \right]^{1/(m-1)} \ldots (3.2.9)
\]

From equation (3.2.1)

\[
\frac{\lambda}{m} = \frac{1}{\sum_{k=1}^{c} 1 / d(x_{j}, y_{k}) - \alpha} \ldots (3.2.10)
\]

and substituting equation (3.2.10) into Equation (3.2.9) we obtain

\[
u_{j} = \left[ \frac{1}{\sum_{k=1}^{c} d(x_{j}, y_{k}) - \alpha / d(x_{j}, y_{k}) - \alpha} \right]^{1/(m-1)} \ldots (3.2.11)
\]
Since $P_j$ is independent of $y_i$, the update equation is the same as in the FCM and it is given in equation (3.2.4).

Note that if $\alpha = 0$, the proposed extension of the FCM reduces to the standard algorithm. Since the terms of the form $d(x_j, y_i) - \alpha$ in equation (3.2.11) may take zero or negative values, resulting in undefined or negative membership values, we should define.

$$ u_{ij} = 1, \text{ for } d(x_j, y_i) \leq \alpha \quad \ldots (3.2.12) $$

The physical interpretation of equation (3.2.12) is to define a hard region around quantization centers with radius $\alpha$. Therefore, any pixel value contained in these hard regions will be classified in a hard manner and those outside the hard regions will go through fuzzy quantization according to the membership function defined in equation (3.2.11). Since no membership value needs to be calculated for those pixels staying in the hard region, the algorithm will work faster depending on the distribution of the colour vectors and the radius of the hard region. The more the pixels contained in the hard region, the faster the quantization. On the other hand, if too many pixels are contained in the hard region, we lose the benefits of the fuzzy quantization and improvement ever hard quantization becomes less significant. A natural solution is to set the radius of the hard regions to a fraction of the distance between the closest two quantization centers.

**Results**

Partitioned Index Maximization (PIM) algorithm is implemented for colour quantization of images. First, we need to determine the value of $\alpha$ used in equation (3.2.6) that controls the amount of contribution of partition index to fuzzy quantization. Equation (3.2.12) dictates that the value of $\alpha$ should not exceed halfway between two clusters. Therefore, it is dependent on the distribution of the cluster centers.

A straightforward approach is to set $\alpha$ to a fraction of the distance between the closest two quantization centers, that is $\alpha = \delta \min[d(y_i, y_j)]$ where
0 \leq \delta < 0.5. Hence, \( \alpha \) is dynamically determined. Thus the percentage change in msc versus \( \delta \) for 100 test images as when \( \delta = 0 \) corresponds to classical fuzzy quantization and as \( \delta \) increases, the hard region around the quantization centers grows. It is seen that the PIM algorithm yields about 10% decrease in msc with respect to FCM for \( m = 1.5 \), while the decrease in msc is close to 5% for \( m = 1.3 \).

It is interesting to see that the percentage decrease in msc is greatly affected by the choice of \( m \). That is, the fuzzier the quantizer is, the more the improvement obtained in msc as \( \delta \) approaches 0.5. The reason can be explained as follows: If a pixel \( x_j \) is in the \( \delta \) region of the quantization colour \( y_i \), there is not much doubt about which quantization center it belongs to. Therefore, PIM algorithm assigns \( x_j \) to \( y_i \) in a hard manner, that is \( u_{ij} = 1 \). If \( x_j \) is outside the \( \delta \) region of all quantization centers, its membership value is calculated by the fuzzy membership function defined in equation (3.2.11). If \( m \) is small, the degree of fuzziness is already low and the effect of introducing a hard region around quantization centers is also small. If \( m \) is larger, the degree of fuzziness is larger and the effect of the hard region defined by \( \delta \) increases. Therefore, better codebooks are obtained by fuzzy quantization with respect to hard quantization, but even better codebooks are obtained by a semi-fuzzy quantization schema.

This result brings out the question of the effect of \( m \) and how \( \delta \) contributes to the resultant msc for colour image quantization. It is seen that the value of \( m \) plays an important role in fuzzy quantization, and our experiments show that setting \( m = 1.3 \) produces satisfactory results for most images. However, introducing a hard region around quantization centers further reduces msc, more for relatively higher \( m \) values. Therefore, quantization process becomes less sensitive to the \( m \) parameter and lower msc Values are obtained for any \( m \) value with respect to the FCM algorithm.
We also measured the error after applying the modulation transfer function (MTF) of the human visual system (HVS) to the resulting images as suggested in [3, 218, 219]. We differentiated the colour image into its luminance and chrominance components by transforming it into YC_bC_r colour coordinate system [152]. Both components of the image are then filtered with the HVS filter, and we computed the frequency weighted mean squared error (FWMSE) afterwards as in Ref. [158].

The images for which the performance is best, worst and average are also indicated. It is to know that there is a perceptual improvement in 90 images for quantization to 16 levels using $m = 1.3$.

The colour differences ($\Delta E$) of the test images in $L^*a^*b^*$ colour coordinate system [90] are also calculated.

$$\Delta E = \left[ (\Delta L^*)^2 + (\Delta a^*)^2 + (\Delta b^*)^2 \right]^{1/2} \quad \text{(3.2.13)}$$

The results are compared for 100 test images. In general, there is a decrease in colour differences with respect to FCM quantization for most images, resulting in perceptually better quantized images.

Introducing a hard region also affects the speed of the quantization. The membership values of the pixels in the hard region need not be calculated, they are simply set to one. If $\delta$ is chosen to be close to 0.5 about 40–50 % of the pixels are classified in a hard manner on the average. This schema has an accelerating effect for most images. The more the pixels contained in the hard region, the faster the algorithm gets.

The percentage increase in relative computation times (RCT) with respect to the FCM algorithm is given for test images. In general, the algorithm converges faster as hard region grows out. The increase in rct per iteration versus $\delta$ is defined as $\delta$ approaches 0.5, each iteration gets about 10% faster than
it is in FCM. The percentage increase in ret per iteration for $\delta = 0.49$ is 8.1% with $\sigma = 2.1$ for 100 test images.

We also implemented the PIM algorithm for clustering of two dimensional test data given in [14]. The data consist of two clusters which are distinctly apart from shows the partition imposed by the FCM algorithm with two clusters for $m = 1.3$. The FCM algorithm tries to partition the feature space into equal size clusters. The classification imposed by the PIM algorithm with $\delta = 0.4$. The PIM algorithm assigns higher membership values to nearby vectors and lower membership values to further vectors. Where the membership functions of both the FCM and the PIM algorithms are sketched in two dimensions. Therefore, a better performance is obtained by the PIM algorithm if the clusters are relatively compact.

Here, fuzzy methodologies are investigated for colour image quantization and the PIM algorithm is introduced which is derived from the FCM algorithm by the inclusion of a term for partition index. The partition index term is used to modify the membership function such that higher membership values are assigned to nearby vectors with respect to codebook entries and lower membership values are assigned to further vectors. The overall effect of the inclusion of the partition index is not only to obtain quantized images luminance and chrominance channels indicating better perceptual quality. Since a hard region is formed around the quantization centers, the speed of the algorithm is also improved over the FCM algorithm is also investigated and an appropriate method to dynamically determine its value is suggested. Although the intended target area of the PIM algorithm is colour image quantization, it is also applicable to the general classification problems. Our experiments show that better codebooks are obtained faster than the FCM algorithm by the PIM algorithm.
3.3. FUZZY SCALAR AND VECTOR MEDIAN FILTERS BASED ON FUZZY DISTANCES

The fuzzy scalar median (FSM) is proposed defined by using ordering of fuzzy numbers based on fuzzy minimum and maximum operations defined by using the extension principle. Alternatively, the FSM is defined from the minimization of fuzzy distance measure, and the equivalence of the two definitions is proven. Then, the fuzzy vector median (FVM) is proposed as an extension of vector median, based on a novel distance definition of fuzzy vectors, which satisfy the property of angle decomposition. By defining properly the fuzziness of a value, the combination of the basic properties of the classical scalar and vector median (VM) filter with other desirable characteristics can be succeeded.

The median operator has been extensively used as a filter in digital signal and image processing [170]. It combines the desirable characteristics of the impulsive noise removal and the preservation of the signal or image edges. One of the most popular techniques to process a multichannel signals corrupted by impulsive noise is the vector median (VM) filter, which inherently utilizes the correlation of the channels and keep the desirable properties of the scalar median, the zero impulse response, and the preservation of the signal edges [7].

Here, the fuzzy scalar median (FSM) and the fuzzy vector median (FVM) are discussed. Fuzziness can describe ambiguity regarding a pixel's chromatic value or location. Which is estimated by the chromatic values of a pixel or a pixels' neighbourhood.

The fuzzy scalar median is defined by minimizing a fuzzy distance function. The angle decomposed fuzzy vectors and fuzzy vector median are presented.
Fig. 3.3.1(a) Two fuzzy numbers \( X_1, X_2 \), and their \( \alpha \)-cuts limits and (b) maximum (solid line) and minimum (dashed line) of two fuzzy numbers.

Fig.3.3.2(a) Five fuzzy numbers \( X_1, X_2, X_3, X_4, X_5 \) and (b) the fuzzy order statistics \( X_{(1)}, X_{(2)}, X_{(3)}, X_{(4)}, X_{(5)} \).
Fuzzy Scalar Median defined through fuzzy ordering

The $\alpha$-cuts of a fuzzy set are the classical sets $X^\alpha$, where

$$\alpha \in X^\alpha \Leftrightarrow \mu(x) \geq \alpha, \alpha \in [0,1].$$

A fuzzy set is called normal if $\exists x: \mu(x) = 1 \text{ or } X^1 \neq \emptyset$. It is called convex if $\forall \alpha_1, \alpha_2 \in [0,1], \alpha_1, > \alpha_2, \Leftrightarrow X^{\alpha_1} \subseteq X^{\alpha_2}$. A normal and convex fuzzy set is called fuzzy number [107, 119, 229]. A one-dimensional fuzzy number is called convex fuzzy number when the corresponding $\alpha$-cuts are convex sets.

The maximum and minimum of two fuzzy numbers $X_1, X_2$, are given [107] by

$$\text{MAX } \{X_1, X_2\} = \bigcup_{\alpha} \{\max \{x^{\alpha}_{1r}, x^{\alpha}_{2r}\}, \max \{x^{\alpha}_{1r}, x^{\alpha}_{2r}\}\} \quad \quad \text{(3.3.1)}$$

$$\text{MIN } \{X_1, X_2\} = \bigcup_{\alpha} \{\min \{x^{\alpha}_{1r}, x^{\alpha}_{2r}\}, \min \{x^{\alpha}_{1r}, x^{\alpha}_{2r}\}\} \quad \quad \text{(3.3.2)}$$

The fuzzy minimum and fuzzy maximum of two fuzzy numbers $X_1, X_2$ are shown in fig.3.1.3(a), and fig.3.2.2(b).

Any order statistic of $n$ fuzzy numbers $X_1, X_2, \ldots X_n$ can be found by using successive fuzzy maximum and minimum operators. The ordered $n$ fuzzy numbers can be calculated by the union of the corresponding ordered $\alpha$-cuts. These ordered $\alpha$-cuts are called the $\alpha$-cuts of the fuzzy numbers. Let the ordered fuzzy numbers be symbolized as $X_{(1)}, X_{(2)}, \ldots X_{(n)}$, where $X_{(1)}$ is the fuzzy minimum and $X_{(n)}$ the fuzzy maximum. The ordered fuzzy numbers are also written as

$$X_{(i)} = \bigcup \{x^{\alpha}_{(i)r}, x^{\alpha}_{(i)l}\}, \quad i = 1, 2, \ldots, n \quad \quad \text{(3.3.3)}$$
Where $x^{a}_{(0)}$ are the ordered statistics of $x^{a}_{(0)}$, and $x^{a}_{(0)}$, are the ordered statistics of $x^{a}_{(0)}$. Similarly, we can call the ordered fuzzy numbers $X_{(1)}, X_{(2)}, ..., X_{(n)}$. Then, the median $M_a$ defined by $n$ fuzzy numbers is

$$M_a \{X_1, X_2, \ldots, X_n\} = U \left[ x^{a}_{(\lfloor\frac{n+1}{2}\rfloor)}, x^{a}_{\lfloor\frac{n+1}{2}\rfloor} \right], \text{ when } n \text{ is odd. ...}(3.3.4)$$

If $n$ is even, the median can be defined by their arithmetic mean $\frac{1}{2}(X_{(n/2)} + X_{(n/2)+1})$. The fuzzy numbers $X_1, X_2, X_3, X_4, X_5$ are illustrated in fig. 3.3.2(a). Fuzzy order statistics $X_{(1)}, X_{(2)}, X_{(3)}, X_{(4)}, X_{(5)}$ are illustrated in fig. 3.3.2(b). Here the fuzzy median $X_{(3)}$ and fuzzy maximum $X_{(5)}$ are equal to $X_3$.

Fuzzy scalar median defined through the minimization of a fuzzy distance

The median of a set of crisp numbers $x_1, x_2, ..., x_n$ is $\sum_{i=1}^{n} |x_i - m|$, based on the $L_1$ distance norm. The norm $d_i[., .]$ that corresponds to the distance of two fuzzy numbers $X$ and $Y$ defined [229] as

$$d_i[X, Y] = \frac{1}{2} \int_{\alpha=0}^{\alpha} \left( |x_i^\alpha - y_i^\alpha| / |x_r^\alpha - y_r^\alpha| \right) d\alpha$$

which is the integral of the distances of the corresponding $\alpha$-cuts. Then, the fuzzy median can be defined by using (3.3.5).

**Definition 3.3.1:** The FSM of $X_1, X_2, ..., X_n$ fuzzy number is the fuzzy number $M = U \left[ m^\alpha, m^\alpha \right]$ which minimizes the expression $E = \sum_{i=1}^{n} d_i[X_i, M]$.

By using relation (3.3.5), the error function $E$ is now a crisp function.

$$E = \int_{\alpha=0}^{\alpha} \sum_{i=1}^{n} \left( |x_i^\alpha - m_i^\alpha| + |x_r^\alpha - m_r^\alpha| \right) d\alpha$$

...(3.3.6)
The sums $E_\alpha = \sum |x_{i\alpha}^a - m_{i\alpha}^a| + |x_{i\alpha}^r - m_{i\alpha}^r|$, the minimization of each $E_\alpha$ is equivalent to the minimization of $E$, since $E_\alpha$ are all positive. The parameters $m_{i\alpha}^a$, $m_{i\alpha}^r$ are determined by crisp minimization[164] as $m_{i\alpha}^a = \text{median}\{x_{i1}^a, x_{i2}^a, ..., x_{in}^a\}$ and $m_{i\alpha}^r = \text{median}\{x_{i1}^r, x_{i2}^r, ..., x_{in}^r\}$, $\forall \alpha \in [0,1]$. Thus, FSM
\[
M = U[\text{median}\{x_{i1}^a, x_{i2}^a, ..., x_{in}^a\}, \text{median}\{x_{i1}^r, x_{i2}^r, ..., x_{in}^r\}]
\]
(3.3.7)
If $n$ is odd, the median of the $n$ crisp lower $x_{i\alpha}^a$ and upper $x_{i\alpha}^r$ limits, are the numbers $x_{i\alpha}^a[{(n+1)/2}]$ and $x_{i\alpha}^r[{(n+1)/2}]$, respectively. Thus, the FSM is equivalent to the previous one, is in relation (3.3.4). If $n$ is even, the number that belong to the intervals $[x_{i(n/2)^a}, x_{i(n/2)^a}]$ and $[x_{i(n/2)^r}, x_{i(n/2)^r}]$ can be considered as the median.

The FSM can be composed by the limits of the $\alpha$ - cuts of more than one fuzzy numbers. Which leads to this result when it is minimized unconditionally.

**Definition 3.3.2:** The FSM of $X_1, X_2, ..., X_n$ fuzzy number is the fuzzy number $X_m$ such that $X_m \in \{X_1, X_2, ..., X_n\}$ and $\forall j = 1, 2, ..., n, j \neq m$.

\[
\sum_{i=1}^{n} D_n[X_m, X_i] < \sum_{i=1}^{n} D_n[X_j, X_i].
\]

**Angle Decomposed Fuzzy Vectors**

Let $X$ be an $n$-dimensional fuzzy set, $\mu_\alpha(x)$ be its membership function of $n$ variables, and $X^\alpha$ the corresponding $\alpha$-cuts, where $x \in X^\alpha \Leftrightarrow \mu(x) \geq \alpha, \alpha \in [0,1]$. Consider the vector $x_c$ when $\mu(x_c) = 1$, is the center of the fuzzy set. Consider also $n-1$ angles $\theta = (\theta_i, i = 1, 2, ..., (n-1)), \theta_i \in [0, \pi)$. The center of the fuzzy set $x_c$ and each angle $\theta$, determine a hyperplane. The intersection of $n-1$ hyperplanes is a straight line (direction) in the $n$-dimensional hyperspace, where a function
\( \mu_1 \) can be defined as \( \mu_1(x, \theta) = \mu_1(x_1(x, \theta), x_2(x, \theta), \ldots, x_{n-1}(x, \theta), x) \). This function can be considered as a membership function of an 1-D fuzzy set \( X^\theta \).

**Definition 3.3.3:** An n-dimensional fuzzy set \( X \) is an angle decomposed fuzzy vector (ADFV), if, for each vector of angles \( \theta = (\theta_1, \theta_2, \ldots, \theta_{n-1}) \), the 1-D fuzzy set \( X^\theta = \{x, \mu_1(x, \theta)\} \) is a convex fuzzy number.

Fig. 3.3.3(a) Two 1-D convex fuzzy numbers \( X^{\theta_1}, Y^{\theta_1} \), coming from two 2-D ADFV \( X, Y \) when the angle vector \( \theta = (\theta_1) \) is determined and (b) the upper \( x_{1\alpha^{\theta_1}}, y_{1\alpha^{\theta_1}} \) and lower \( x_{10^{\theta_1}}, y_{10^{\theta_1}} \) limits of two \( \theta \alpha \)-cuts \( X^{\theta \alpha}, Y^{\theta \alpha} \) the centers of the ADFV \( x, y \) and the distance between them.

Fig. 3.3.4(a) Distance between two 2-D ADFV \( X, Y \) with elliptical \( \alpha \)-cuts and axes \( f_{1\alpha}, f_{2\alpha} \) and \( f_{1\alpha'}, f_{2\alpha'} \), respectively, which are reduced linearly from their maximum values \( f_{1\alpha}, f_{2\alpha}, f_{1\alpha'}, f_{2\alpha'} \) for \( \alpha = 0 \), to zero for \( \alpha = 1 \). (b) The distance of the centers is 100, \( f_{1\alpha} \) and \( f_{2\alpha} \) vary from zero to 50, and \( f_{1\alpha'} = 10, f_{2\alpha'} = 30 \).

A two-dimensional (ADFV) and the angle decomposed 1-D convex fuzzy number is shown in fig. 3.3.3(a). It can also prove that if \( \theta \) is a \((n - 1 - k)\) dimensional vector, the function \( \mu_k(x_1, x_2, \ldots, x_k, \theta) \) can be considered as a membership function of a \( k \)-dimensional ADFV. Then the ADFV will be defined as
Where \( x^\alpha, y^\alpha \), and \( x^\alpha, y^\alpha \) are lower and upper points that limit the \( \alpha \)-cuts of the corresponding 1-D \( X^\theta \) fuzzy numbers. When the Euclidean norm define a distance between classical n-dimensional vectors \( x = (x_1, x_2, \ldots, x_n) \) and \( y = (y_1, y_2, \ldots, y_n) \) as

\[
d_l(x, y) = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + \ldots + (x_n - y_n)^2}
\]  

... (3.3.9)

the Euclidean fuzzy distance can be defined by using (3.3.8) and (3.3.9). When the fuzzy vectors are described by using \( \alpha \)-cuts, for a given and a vector of angles \( \theta = (\theta_1, \theta_2, \ldots, \theta_{n-1}) \), two points \( x^\alpha, y^\alpha \) and \( x^\alpha, y^\alpha \) are defined, which are the lower and the upper limits of the corresponding \( \theta \) \( \alpha \)-cut. The proposed Euclidean fuzzy distance is the normalized integral of all the distances \( d^2_e(x^\alpha, y^\alpha) \) between the lower limits, and the distances \( d^2_e(x^\alpha, y^\alpha) \) between the upper limits, for every \( \alpha \in [0,1] \) and \( \theta_i \in [0, \pi], i = 1, 2, \ldots, n-1 \).

Then, the distances between two lower and two upper limits are equal to

\[
d^2_e(x^\alpha, y^\alpha) = (d^\alpha_{lx} - d^\alpha_{lx})^2 + (d^\alpha_{ly} - d^\alpha_{ly})^2 \prod_{i=1}^{n-1} \cos(\theta_i) + d^2 xy \]  

... (3.3.10)

\[
d^2_e(x^\alpha, y^\alpha) = (d^\alpha_{lx} - d^\alpha_{lx})^2 - 2(d^\alpha_{lx} - d^\alpha_{ly}) \prod_{i=1}^{n-1} \cos(\theta_i) + d^2 xy \]  

... (3.3.11)

where \( \theta_i \), \( i = 1, 2, \ldots, n - 1 \) are known angles \( \theta_i \in [0, \pi) \). By using (3.3.8), (3.3.10) and (3.3.11), the Euclidean fuzzy distance between two ADFV \( X, Y \) is given by

\[
D_{ex}[X, Y] = d^2 xy + d^2 xy
\]

... (3.3.12)
Thus the Euclidean fuzzy distance can be considered as a generalized Euclidean distance.

**Fuzzy Vector Median definition and properties**

**Definition 3.3.4:** The fuzzy vector median (FVM) of $X_1, X_2, \ldots, X_n$ ADFV is the ADFV $X_{FVM}$ such that $X_{FVM} \in \{X_i, i = 1,2,\ldots,n\} \text{ and } \forall j = 1,2,\ldots,n, j \neq k : \sum_{i=1}^{n} D_n[X_{FVM}, X_i] < \sum_{i=1}^{n} D_n[X_j, X_i]$ The Euclidean fuzzy distance is used as the Euclidean FVM $X_{FVM}$ as

$$S^n = \sum_{i=1}^{n} D_n[X_i,Y]$$

it minimizes the same expression, when Y should be one of $X_i$. As a result, the FVM of a set of fuzzy vectors is affected by the presence of fuzziness.

Let us symbolize by $d_y$ the distance of the centers of two ADFV $X_i, X_j$ and $d_{fj}$ the distance depending on the fuzziness given by

$$S^n = \sum_{j=1}^{n} d_y = \sum_{j=1}^{n} d^2(y, x_j)$$

Without loss of generality, it is assumed that $S_i < S_{i+1}, \forall i = 1,2,\ldots,n-1$, which means that $x_{a_i}$ is the classical VM of the centers. Thus,

$$S_{e**(k)} - S_{a_i} = C_{**(k)} > 0, k = 1,2,\ldots,n-i.$$  

The ADFV $X_{i+k}$ will be the FVM if and only if $S_{e**(k)} < S_{e**}, j = 1,2,\ldots,n, j \neq i + k$. the above condition is equivalent to

$$\sum_{j=1}^{n} d_{f**(k)} < \sum_{j=1}^{n} d_{fj} - C_{**(k)}.$$  

Two alternative definitions of the fuzzy scalar median and fuzzy numbers based on the extensions of the minimum and maximum operations through the extension principle, fuzzy distance measure were proven. Then, the
fuzzy vector medians based on a novel distance definition of fuzzy vectors were also investigated.

3.4. FUZZY CLUSTERING ALGORITHMS BASED ON RESOLUTION AND THEIR APPLICATION IN IMAGE COMPRESSION

Cluster analysis refers to a broad spectrum of methods which try to subdivide a data set $X$ into subsets or clusters, which are pair-wise disjoint, all non-empty, and reproduce $X$ via union. The clusters are termed as a hard (i.e., non-fuzzy) $c$-partition of $X$. A significant fact based on the idea is the defect in the underlying axiomatic model that each point in $X$ is unequivocally grouped with other members of “its” cluster, and thus bear no apparent similarity to other members of $X$. Fuzzy sets characterize an individual point’s similarity to all the clusters was introduced in 1965 by Zadeh as a new way to represent vagueness in everyday life [248]. The use of fuzzy sets in clustering was first proposed [13] and several classification schemes were subsequently developed [51]. Bezdek extended Dunn’s formulation and produced a family of infinitely many fuzzy $c$-means, which includes Dunn’s original algorithm as a special case [15, 71].

Clustering or codebook design is an essential process in image compression based on vector quantization. In the context of vector quantization, the clustering process is also referred to as the codebook design. In a codebook, each block of the image can be represented by the binary address of its closest codebook vector. Such a strategy results in significant reduction of the information involved in by replacing each image block by its closest codebook vector. As a result, the quality of the reconstructed image strongly depends on the codebook design.

Codebook design is usually based on the minimization of an average distortion measure between the training vectors and the codebook vectors [104]. The minimization of the distortion measure is widely performed by a gradient
descent-based iterative algorithm that is known as c-means or Generalized Lloyd algorithm. Although the c-means algorithm is simple and intuitively appealing, it strongly depends on the selection of the initial codebook, and it can be trapped in local minima.

The techniques for codebook design mentioned above are based on hard decisions in the sense that each training vector is assigned to a single cluster according to some criterion. Since fuzzy clustering algorithms are a kind of the partition of the training vector space based on soft decisions, they provide a new way to resolve the problems in image compression based on vector quantization encountered by those based on hard decisions. The fuzziness of fuzzy clustering algorithms can be used to eliminate, or at least significantly reduce, the dependence of the resulting codebook on the selection of the initial codebook. Bezdek’s family of fuzzy c-means is developed for vector quantization. However, vector quantization is based on the representation of any image block by a single codebook vector, which is essentially a hard decision-making process. In addition, the existing fuzzy algorithms require a priori assumptions about the level of fuzziness appropriate for a certain clustering task. Such assumptions are implicitly made by selecting any of the fuzzy c-means algorithms. Furthermore, these fuzzy algorithms are computationally expensive so that they are in practical in the engineering. Fuzzy vector quantization algorithms presented by Karayiannis and Pin-I Pai are the improvement of Bezdek’s algorithms by effective strategies for the transition from soft decisions to hard decisions. But the algorithms are intuitive and lack theoretical preciseness.

Here we presents an idea of clustering resolution and deduces fuzzy clustering algorithms based on the idea. The presented clustering algorithms can not only overcome the effect of the initial codebook easily, but also achieve a better tradeoff between the convergent speed and the quality of clustering.
An idea of clustering resolution

**Definition 3.4.1.** Let \( X = \{x_1, x_2, ..., x_n\} \) be a set of \( n \) observations in \( \mathbb{R}^r \) (\( r \)-dimensional Euclidean space); \( x_k \) is the \( k^{th} \) training vector, \( x_{k,j} \) is the \( j^{th} \) feature of \( x_k \). Let \( U \) be a real \( c \times n \) matrix, \( U = [u_{i,k}] \) is the set of memberships. Let \( V \) be a set of cluster centers, \( V = (v_1, v_2, ..., v_c) \in \mathbb{R}^c, v_i \in \mathbb{R}^r, 2 \leq c < n \), \( d_{i,k}^2 \) the distance between the training vector \( x_k \) and the cluster center \( v_i \). That is, \( d_{i,k}^2 = \|x_k - v_i\|^2 \). \( V_{cn} \) is the set of real \( c \times n \) matrices; \( M_{fc} \) is the space of membership function. That is,

\[
M_{fc} = \{ U \in V_{cn} | u_{i,k} \in [0,1], k; \sum_{i=1}^{c} u_{i,k} = 1 \forall k; 0 < \sum_{k=1}^{n} u_{i,k} < n \forall i \}.
\]

**Definition 3.4.2.** (Fuzzy objective function). Let \( J_m: M_{fc} \times \mathbb{R}^c \rightarrow \mathbb{R}^+ \)

\[
J_m(U, v) = \sum_{k=1}^{n} \sum_{i=1}^{c} u_{i,k}^m d_{i,k}^2 \tag{3.4.1}
\]

Where \( U \in M_{fc}, m \in (0, +\infty) \).

**Definition 3.4.3.** (Distance spectrum). Let \( x_i \) be an arbitrary sample of the training vectors, and if Euclidean distances between \( x_i \) and all cluster centers are ordered from the smallest to the largest, the distance sequence is called as the Distance Spectrum of the sample \( x_i \). In terms of distance spectrum, the more the difference from the neighbouring cluster distances on distance spectrum is, the easier is the clustering convergence.

**Definition 3.4.4** (Clustering resolution). The ratio of two memberships of the training sample \( x_k \) with regard to the cluster center \( v_i \) and \( v_j \), that is, \( R \in [1, \infty) \). The bigger the \( R \) is, the more easily \( x_k \) matches \( v_i \) or \( v_j \), and the faster the convergent speed of clustering is, vice versa.

From the foregoing it is clear that distance spectrum and clustering resolution can help analyze clustering more intuitively. On the whole, the membership function of clustering is inversely proportional to the distances.
between the cluster and the training sample. The expression between the cluster distance and its resolution may be

\[ R = \frac{u_{i,k}}{u_{j,k}} = \left( \frac{d_{i,k}^2}{d_{j,k}^2} \right)^w, \]  

...(3.4.2)

where \( w \) is a resolution variable, \( w \in (0, +\infty) \).

**An application of the idea in fuzzy clustering**

**Theorem 3.4.1** (Membership function based on clustering resolution). If equation (3.4.2) is true, and \( \sum_{j=1}^{c} u_{j,k} = 1 \), with \( w \in (0, +\infty) \), \( \forall k \), then

\[ u_{i,k} = \frac{1}{\sum_{j=1}^{c} \left( \frac{d_{i,k}^2}{d_{j,k}^2} \right)^w} \]  

...(3.4.3)

**Proof:** In equation (3.4.2) let \( j = i+1 \), then

\[ u_{i+1,k} = \left( \frac{d_{i,k}^2}{d_{i+1,k}^2} \right)^w u_{i,k}, \]  

...(3.4.4)

Using recursion and simplifying it, thus

\[ u_{i,k} = \left( \frac{d_{i,k}^2}{d_{i,k}^2} \right)^w u_{i,k} \]  

...(3.4.5)

and since

\[ \sum_{j=1}^{c} u_{j,k} = \sum_{j=1}^{c} \left( \frac{d_{1,k}^2}{d_{j,k}^2} \right) u_{1,k} = 1 \]  

...(3.4.6)

Simplifying equation (3.4.6)

\[ u_{1,k} d_{1,k}^{2w} = \frac{1}{\sum_{j=1}^{c} \left( \frac{1}{d_{j,k}^2} \right)^w} \]  

...(3.4.7)

In equation (3.4.5) using equation (3.4.7) and simplifying it

\[ u_{i,k} = \frac{1}{\sum_{j=1}^{c} \left( \frac{d_{i,k}^2}{d_{j,k}^2} \right)^w}, \]  

...(3.4.8)
Result: (Fuzzy c-means algorithm). Fix $m \in (1, +\infty)$, let $X$ have at least $c<n$ distinct points, and define $\forall k$ the set $I_k = \{i | 1 \leq i \leq c; d_{i,k}^2 = \|x_i - v_k\|^2 = 0\}$,

$I_k = \{1,2,\ldots,c\} \setminus I_k$, then $(U,v) \in \mathcal{M}_c \times R^c$ may be globally minimal for $J_m$ in equation (3.4.1) only.

If $I_k=\emptyset$, then equation (3.4.3) is true.

If $I_k=\emptyset$, then $u_{i,k} = 0$, $\forall i \in I_k$, and $\sum_{i \in I_k} u_{i,k} = 1$

and

$$v_i = \frac{\sum_{k=1}^n x_k u_{i,k}^m}{\sum_{k=1}^n u_{i,k}^m} \text{ with } m=1+\frac{1}{w} \quad \quad \text{...}(3.4.9)$$

**Theorem 3.4.2** (Fuzzy clustering based on resolution; FCR)

Let $I_k = \{i | 1 \leq i \leq c; d_{i,k}^2 = \|x_i - v_k\|^2 = 0\}$, $I_k = \{1,2,\ldots,c\} \setminus I_k$.

If $I_k=\emptyset$ then equation (3.4.3) is true.

Or

If $I_k \neq \emptyset$, then $u_{i,k} = 0$, $\forall i \in I_k$ and $\sum_{i \in I_k} u_{i,k} = 1$;

$J_m$ in Equation (3.4.1) may be globally minimal only if

$$v_i = \frac{\sum_{k=1}^n x_k u_{i,k}^m A(i,k)}{\sum_{k=1}^n u_{i,k}^m A(i,k)} \quad \quad \quad \text{...}(3.4.10)$$

where

$$A(i,k) = \left\{1 + wm \left(\sum_{i=1}^c u_{i,k} \left(\frac{d_{i,k}^2}{d_{i,k}^2 + \alpha \|x_i - v_k\|^2}\right) - 1\right)\right\},$$

where $\alpha = w+1-wm$. 


Proof: \( I_k = \emptyset \).

In equation (3.4.1) using equation (3.4.3) and resolving equation (3.4.1),

\[
J_m = \sum_{k=1}^{n} \left[ \frac{d^2_{i,k}}{d^2_{j,k}} \right] + \sum_{l=1}^{c} \left[ \frac{d^2_{i,k}}{d^2_{j,k}} \right]
\]

\[
= \sum_{k=1}^{n} \frac{d^2_{i,k}}{\sum_{j=1}^{c} \left( d^2_{i,k} / d^2_{j,k} \right)^w} + \sum_{l=1}^{c} \left[ \sum_{j=1}^{c} \left( d^2_{i,k} / d^2_{j,k} \right)^w \right]
\]

\[
= J_{m1} + J_{m2}
\]  

...(3.4.11)

Letting directional derivative in equation (3.4.11) equal to zero, yields

\[
\frac{dJ_m}{dv_i} = \frac{dJ_{m1}}{dv_1} + \frac{dJ_{m2}}{dv_1} - 0,
\]

...(3.4.12)

\[
\frac{dJ_{m1}}{dv_1} = \sum_{k=1}^{n} \left[ -2(x_k - v_i) \right] u_{i,k} \left( 1 - wm + wmu_{i,k} \right)
\]

...(3.4.13)

\[
\frac{dJ_{m2}}{dv_1} = \sum_{k=1}^{n} \sum_{l=1}^{c} \left[ -2(x_k - v_i) \right] wmu_{i,k} u_{i,k} \left( \frac{d^2_{i,k}}{d^2_{j,k}} \right)^{1+w-wm}
\]

...(3.4.14)

\[
\frac{dJ_m}{dv_i} = \sum_{k=1}^{n} -2(x_k - v_i) u_{i,k} \left[ 1 - wm + wmu_{i,k} +wm \sum_{l=1}^{c} u_{i,k} \left( \frac{d^2_{i,k}}{d^2_{j,k}} \right)^{1+w-wm} \right] = 0
\]

...(3.4.15)

Simplifying equation (3.4.15)

\[
\frac{dJ_m}{dv_i} = \sum_{k=1}^{n} -2(x_k - v_i) u_{i,k} \left[ 1 - wm + wmu_{i,k} +wm \sum_{l=1}^{c} u_{i,k} \left( \frac{d^2_{i,k}}{d^2_{j,k}} \right)^{1+w-wm} \right] = 0
\]

...(3.4.16)

Where \( I_k \neq \emptyset \), then \( u_{i,k} = 0, \forall i \in I_k \) and \( \sum_{i \in I_k} u_{i,k} = 1 \) (see[15] for a detailed account).

Deductions 3.4.3. (i) when \( \alpha \) is zero, i.e., \( w=1/(m-1) \). Theorem 3.4.2 is Fuzzy c-Means Algorithm:
(ii) when \( w=1/m \) is true. Theorem 3.4.2 is also Fuzzy c-Means Algorithm;

(iii) (i) is equivalent to (ii), i.e, in Fuzzy c-Means Algorithm \( m \) is generalized to 
\((0,\infty)\)

(iv) \( w \to \infty \), Theorem 3.4.2 becomes c-Means Algorithm;

**Proof**

(i) Using \( w=1/(m-1) \) in equation (3.4.10), noting that \( \sum_{i=1}^{c} u_{i,k} = 1 \) and \( m = 1 + \frac{1}{w} \) hold, then equation (3.4.4) becomes

\[
\nu_i = \frac{\sum_{k=1}^{n} x_{i,k} u_{i,k}^{1+1/w}}{\sum_{k=1}^{n} u_{i,k}^{1+1/w}}, \quad \cdots (3.4.17)
\]

So when \( \alpha \) is zero, according to equations (3.4.9) and (3.4.17) theorem 3.4.2 is Fuzzy c-Means Algorithm;

(ii) When \( w=1/m \) is true, using \( w^m=1 \) in equation (3.4.10), then \( A(i,k) \) becomes

\[
A(i,k) = \left[ 1 + \frac{\sum_{l=1}^{c} u_{i,l} \left[ \frac{d_{i,k}^2}{d_{l,k}^2} \right]^w}{\sum_{l=1}^{c} u_{i,l} \left[ \frac{d_{i,k}^2}{d_{l,k}^2} \right]^w} - 1 \right] \quad \cdots (3.4.18)
\]

from equation (3.4.2) \( u_{i,k} = u_{i,k} \left( \frac{d_{i,k}^2}{d_{l,k}^2} \right)^w \), then \( A(i,k)=c_{u_i} \) and using \( m=1/w \), equations (3.4.10) becomes equation (3.4.9), that is, Theorem 3.4.2 is also Fuzzy c-Means Algorithm;

(iii) From both \( w=1/(m-1) \) and \( w=1/m \), we can deduce equation (3.4.9) and both have the same membership function. So both are equivalent. In m-w coordinate plane, \( m=1/w \) is the coordinate scale of \( w=1/(m-1) \), which provides a one-to-one correspondence between both. Thus in Fuzzy c-Means Algorithm \( m \) is generalized to \((0,\infty)\).

(iv) Let \( S = \{s_1, s_2, \ldots, s_n\}, s_i \in \mathbb{N}^k \forall i = 1, 2, \ldots, n; \) \( s_i \) is the index set of the distance spectrum of the \( i^\text{th} \) training sample. Then

\[
d_{s_{i,j},t}^2 \leq d_{s_{2,i},t}^2, \ldots, d_{s_{n,i},t}^2, \forall i = 1, 2, \ldots, n \quad \cdots (3.4.19)
\]

From equation (3.4.19) if see that \( s_{i,1} \) is the index of the nearest cluster center from the \( i^\text{th} \) training sample. If \( w \to \infty \) equation (3.4.3) becomes
and equation (3.4.10) becomes

\[ v_i = \frac{\sum_{k=1}^{n} x_k u_{i,k}}{\sum_{k=1}^{n} u_{i,k}} \quad \ldots \quad (3.4.21) \]

We know from equations (3.4.20) and (3.4.21) that if \( w \to \infty \), Theorem 3.4.2 becomes c-Means Algorithm:

**Resolution analysis of fuzzy objective function**

According to clustering resolution \( R \) in Definition (3.4.1) becomes

\[ J_m = \sum_{I=1}^{n} \left\{ u_{i,k}^m d_{i,k}^2 + \sum_{l=1}^{c} u_{i,k}^m \left[ \frac{d_{i,k}^2}{d_{l,k}^2} \right]^{\gamma w} \right\} \quad \forall I = 1, 2, \ldots, c \quad \ldots \quad (3.4.22) \]

Since equation (3.4.2) gives \( u_{i,k} = u_i \left( \frac{d_{i,k}^2}{d_{l,k}^2} \right)^w \), equation (3.4.22) becomes

\[ J_m = \sum_{k=1}^{n} \left\{ u_{i,k}^m d_{i,k}^2 + \sum_{l=1}^{c} u_{i,k}^m \left[ \frac{d_{i,k}^2}{d_{l,k}^2} \right]^{\gamma w-1} \frac{d_{i,k}^2}{d_{l,k}^2} \right\} \]

\[ = \sum_{k=1}^{n} \left\{ u_{i,k}^m d_{i,k}^2 + \sum_{l=1}^{c} u_{i,k}^m \left[ \frac{d_{i,k}^2}{d_{l,k}^2} \right]^{\gamma w-1} \right\} \]

\[ = \sum_{k=1}^{n} u_{i,k}^m d_{i,k}^2 \left\{ 1 + \sum_{l=1}^{c} \left[ \frac{d_{i,k}^2}{d_{l,k}^2} \right]^{\gamma w-1} \right\} \]

\[ = \sum_{k=1}^{n} u_{i,k}^m d_{i,k}^2 \sum_{l=1}^{c} \left[ \frac{d_{i,k}^2}{d_{l,k}^2} \right]^{\gamma w-1} \quad \ldots \quad (3.4.23) \]

If \( w=1/m \); equation (3.4.23) becomes

\[ J_m = c \sum_{k=1}^{n} u_{i,k}^m d_{i,k}^2 = c \sum_{k=1}^{n} u_{i,k}^{1/w} d_{i,k}^2 \quad \ldots \quad (3.4.24) \]
If \( w = 1/(m-1) \), that is, \( w^{-1} = w \) or \( m-1 = 1/w \), equation (3.4.24) becomes

\[
J_m = \sum_{k=1}^{n} u_{i,k}^m d_{i,k}^2 \sum_{l=1}^{c} \left[ \frac{d_{i,k}^2}{d_{l,k}^2} \right]^w \\
= \sum_{i=1}^{n} u_{i,k}^{m-1} d_{i,k}^2 \\
= \sum_{i=1}^{n} u_{i,k}^{1/\omega} d_{i,k}^2 \quad \ldots (3.4.25)
\]

Comparing equation (3.4.24) with equation (3.4.25), it is clear that the clustering results of \( w = 1/(m-1) \) and \( w = 1/m \) are equivalent, which prove Deductions (c) again.

According to \( s \) in equation (3.4.19), let \( t = s_{i,k} \) in equation (3.4.1), then

\[
J_m = \sum_{k=1}^{n} u_{i,k}^m \cdot k d_{s_{i,k},k}^2 \sum_{l=1}^{c} \left[ \frac{d_{s_{i,k},k}^2}{d_{l,k}^2} \right]^{mw-1} \\
\quad \ldots (3.4.26)
\]

If \( w \to \infty \), using equation (3.4.20) in equation (3.4.1), then we have

\[
J_m = \sum_{k=1}^{n} d_{s_{i,k},k}^2 \cdot k \\
\quad \ldots (3.4.27)
\]

**Fuzzy clustering based on multiresolution**

If the resolution variable \( w \) in \( R \) varies, fuzzy clustering algorithm in theorem 3.4.2 becomes a family based on resolution. When \( w \) is assumed be a continuous function, equations (3.4.3) and (3.4.10) comprise fuzzy clustering algorithm based on multiresolution (FCMR) with regard to the objective function (3.4.1).

**Fuzzy clustering algorithm based on multi resolution’s application in image compression**

Clustering algorithm have broad applications in many realms. In image compression they are used to search code book vectors for vector quantization.
Since image data are enormous, conventional clustering algorithms are difficult to satisfy practical applications. In vector quantization, the sets of training vectors $X$ and codebook vectors $V$ are defined in Definition 3.4.1; the average distortion $D$ is defined as

$$D = \frac{1}{n} \sum_{k=1}^{n} d_{\min}(x_k) = \frac{1}{n} \sum_{k=1}^{n} \min_{j \in C} d_{j,k}^2$$ \quad \text{(3.4.28)}$$

From above it is clear that only the difference between vector quantization and clustering algorithms is the objective function ($D$ can also be regarded as an objective function). According to $S$ in equation (3.4.19), $D$ can be represented as

$$D = \frac{1}{n} \sum_{k=1}^{n} d_{s,k,k}^2$$ \quad \text{(3.4.29)}$$

Comparing equation (3.4.27) with equation (3.4.29), $w \to \infty$, $R \to \infty$, the objective function $J_m$ of FCMR (i.e, c-Means Algorithm) is just the average distortion $D$. Just then, the algorithm, given in appropriate initial codebook, can search a globally optimum codebook very quickly. However, c-Means Algorithm is very sensitive to the initial codebook. If $w$ is small and $R$ is also small, FCMR is very fuzzy, which can overcome the effect of the initial codebook. So when the algorithm begins, we can let $w$ be very small, that is, $R$ very low, so that the fuzziness of FCMR can reduce the dependence of the resulting codebook on the selection of the initial codebook. Next, $w$ becoming big, $m$ decreasing and $R$ increasing equations (3.4.25) and (3.4.26) will be close to (3.4.29), which makes FCMR meet the need of vector quantization. On the course of convergence, $w$ can be controlled by a curve called as resolution control curve.

In practical application, we can take two measures to speed up the algorithm. First, in computing $u_{i,k}, \forall k$, compute $u_{i,k}$ firstly by equation (3.4.3), and then by equation (3.4.5) compute $u_{i,k}$, $i$ belonging to others. Second, when on the resolution control curve $w$ becomes very big the algorithm is very close
to c-Means Algorithm. So in order to simplify computation we can set a transition T and after w is bigger than T, the algorithm becomes c-Means algorithm directly.

To sum up the FCMR algorithm not only reduce the dependence of the resulting codebook on the selection of the initial codebook, but also guarantee the quality of convergence because the fuzzy objective function $J_m$ is gradually close to the average distortion D. Furthermore, the convergent speed of FCMR is by far faster than conventional algorithms, for FCMR is very close to c-Means Algorithm when w becomes very big.

This section presents an idea of clustering resolution based on the idea; fuzzy clustering algorithms based on resolution (FCR) are deduced. These algorithms comprise a set of clustering algorithms; c-means algorithm and fuzzy c-means algorithm are actually special cases in the set. As an application for codebook design in image compression based on vector quantization, fuzzy clustering algorithms based on multiresolution (FCMR) are developed, which easily eliminate the effect of the initial codebook selection on the quality of clustering and also avoid a priori assumption regarding the level of fuzziness necessary for a given clustering task. At the same time, FCMR algorithm achieves a better tradeoff between the convergent speed and the quality of clustering. Furthermore, the range of potential applications of the proposed algorithms is very wide, essentially including any problem involving clustering of random data.