

# Approximate Data Mining in Very Large Relational Data

James C. Bezdek<sup>1</sup>, Richard J. Hathaway<sup>2</sup>,  
Christopher Leckie<sup>3</sup>, Ramamohanarao Kotagiri<sup>3</sup>

<sup>1</sup>Department of Computer Science, University of West Florida, Pensacola, FL 32514, USA

<sup>2</sup>Department of Mathematical Sciences, Georgia Southern University, Statesboro, GA 30460, USA

<sup>3</sup>Department of Computer Science and Software Engineering, University of Melbourne, Victoria, 3010, Australia  
jbezdek@uwf.edu, r.hathaway@ieee.org, caleckie@cs.mu.oz.au, rao@cs.mu.oz.au

## Abstract

In this paper we discuss **eNERF**, an extended version of *non-Euclidean relational fuzzy c-means (NERFCM)* for approximate clustering in very large (unloadable) relational data. The **eNERF** procedure consists of four parts: (i) selection of distinguished features by algorithm **DF** to be monitored during progressive sampling; (ii) progressively sampling a square  $N \times N$  relation matrix  $R_N$  by algorithm **PS** until an  $n \times n$  sample relation  $R_n$  passes a goodness of fit test; (iii) Clustering  $R_n$  using algorithm **LNERF**; and (iv), extension of the LNERF results to  $R_N - R_n$  by algorithm **xNERF**, which uses an iterative procedure based on LNERF to compute fuzzy membership values for all of the objects remaining after LNERF clustering of the accepted sample. Three of the four algorithms are new - only LNERF (called NERFCM in the original literature) precedes this article.

**Keywords:** Cluster analysis, data mining, very large data, non-Euclidean relational fuzzy c-means, progressive sampling, relational data, gene product similarities.

## 1 Introduction

According to Huber (1996), who defines large data sets as an order of magnitude of  $10^8$  bytes, "Some simple standard database management tasks with computational complexity  $O(n)$  or  $O(n \log n)$  remain feasible beyond terabyte monster sets, while others (e.g., clustering) blow up already near large data sets." The next generation of clustering algorithms must not "blow up" when handling large or even very large data ( $\gg 10^{12}$  bytes).

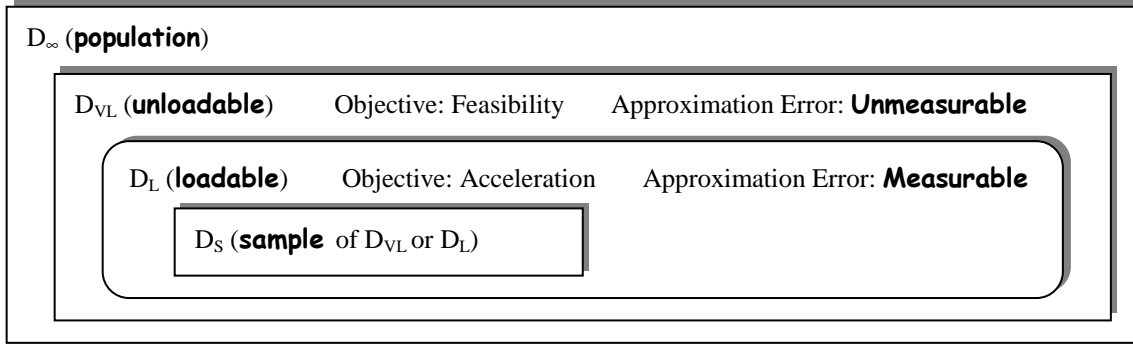
Consider a set of  $N$  objects  $\{o_1, \dots, o_N\}$ . Numerical *object* data has the form  $X = \{\mathbf{x}_1, \dots, \mathbf{x}_N\} \subset \mathcal{R}^s$ , where the coordinates of  $\mathbf{x}_i$  provide feature values (e.g., weight, length, etc.) describing object  $o_i$ . The other type of data commonly found in data mining is numerical *relational* data, which consists of  $N^2$  pair-wise dissimilarities (or similarities), represented by a matrix  $D = [d_{ij} = \text{dissimilarity}(o_i, o_j) \mid 1 \leq i, j \leq N]$ . Many clustering algorithms are known and used for both kinds of data

(Bezdek et al., 1999).  $X$  can be converted into *dissimilarity* data  $D = D(X)$  by computing  $d_{ij} = \|\mathbf{x}_i - \mathbf{x}_j\|$

in any vector norm on  $\mathcal{R}^s$ , so most relational clustering algorithms are (implicitly) applicable to object data. However, there are both similarity and dissimilarity relational data sets that do not begin as object data, and for these, we have no choice but to use a relational algorithm. In many application domains, a relational representation may reflect the way the data is collected and stored. Consider the problems of clustering actors based on whether they have performed together in the same film, or documents based on whether they have similar word usage (Tasker et al., 2001). In each case, a natural representation for the data is as a binary relation that specifies the similarity between objects. A similar problem arises in marketing, when trying to group product lines that are frequently purchased together. Rather than storing every purchasing transaction, a more efficient representation is to simply record the frequency with which individual pairs of products have been purchased together.

Many algorithms have been proposed for clustering in VL object data (Ng and Han, 1994, Bradley et al. 1998, Ganti et al. 1999, Hathaway and Bezdek, 2005), but no algorithms (that we are aware of) exist for "pure" relational data (i.e.,  $D \neq D(X)$  for some  $X$ ). One way to attack the problem of clustering in VL data is discussed in Pal and Bezdek (2002), where the technical notion of *extensibility* is introduced. Roughly speaking, an *extended clustering scheme* applies a clustering algorithm to a (loadable) sample of the full data set, and then non-iteratively extends the sample result to obtain (approximate) clusters for the remaining data. A *literal* scheme applies the clustering algorithm without modification to the full data set.

When the data set is very large (VL), sampling and extension makes clustering feasible for cases where it is not otherwise possible. If the data set is merely large (L), but still loadable, then an extended scheme may offer an approximate solution comparable to the literal solution at a significantly reduced computational cost - in other words, it accelerates the corresponding literal scheme. The benefits for the two cases can be summarized as *feasibility* for VL data sets and *acceleration* for L data sets. Both situations are depicted in Figure 1 where the data set to be clustered is either  $D_L$  or  $D_{VL}$ .



**Figure 1: Population  $D_\infty$  and samples  $D_{VL}$ ,  $D_L$ ,  $D_S$**

Our test for judging whether  $D_S$  is representative of the source may require some basic processing of the full sample. We can load  $D_L$  into primary processing and do the required processing (to test the sample). We cannot load  $D_{VL}$ , but may have to page through  $D_{VL}$  *once* to gather simple statistics (e.g., bin counts for a histogram) needed to assess the quality of candidate samples.

For  $D_L$ , we can assess the approximation error by measuring the difference between the clustering solutions obtained using the corresponding extended and literal schemes. On the other hand, the *only* solution generally available for  $D_{VL}$  is that obtained by the extended scheme, in which case the approximation error cannot be measured. Thus, our confidence in the accuracy of extended clusters in the unverifiable case ( $D_{VL}$ ) is necessarily derived from the verified good behavior we can observe by conducting various  $D_L$  experiments.

## 2 eNERF

**2.1 Labels and Partitions.** Let  $c$  be the number of classes,  $1 < c < n$ . *Crisp label vectors* in  $\mathfrak{R}^c$  look like  $\mathbf{y}_i = (0, \dots, 1, \dots, 0)^T$ , with a 1 in the  $i^{\text{th}}$  place meaning that objects with this label belong to class  $i$ . Fuzzy or probabilistic label vectors ("soft" labels) look like  $\mathbf{y} = (0.1, 0.6, 0.3)^T$  (where  $c=3$  classes, for example); they have entries in  $[0, 1]$  that sum to 1. We need names for the sets of all soft and hard label vectors, so we follow the usual notation:

$$N_{fc} = \{\mathbf{y} = (y_1, \dots, y_c)^T \in \mathfrak{R}^c : \sum y_i = 1; 0 \leq y_i \leq 1 \forall i\};$$

$$N_{hc} = \{\mathbf{y} \in N_{fc} : y_i \in \{0, 1\} \forall i\} \subset N_{fc}.$$

Assume that  $D_N = D = [d_{ij}]$  satisfies, for  $1 \leq i, j \leq N$ ,

$$d_{ij} \geq 0; d_{ij} = d_{ji}; d_{ii} = 0. \quad (1)$$

We do *not* assume that  $D$  is transitive, because this restriction simply does not hold for most real VL relational data. Clustering in  $O$  (or somewhat sloppily, in  $D$ ) is the assignment of (hard or fuzzy or probabilistic) label vectors to the objects in  $O$ . A  $c$ -partition of  $O$  (or  $D$ ) is a set of  $(cN)$  values  $\{u_{ik}\}$  arrayed as a  $c \times N$  matrix  $U = [U^1 \ U^2 \ \dots \ U^N] = [u_{ik}]$ , where  $U^k$ , the  $k^{\text{th}}$  column of  $U$ , is the label vector in  $N_{fc}$  for  $o_k$ . The  $ik^{\text{th}}$  element  $u_{ik}$  of  $U$  is the membership of  $o_k$  in cluster  $i$ . Each column of  $U$  sums to 1, and each row of  $U$  must have at least one non-zero entry.

Table 1 contains an example of crisp ( $U_1$ ) and soft ( $U_2$ ) 2-partitions of  $n=6$  objects.  $U_1$  identifies 3 objects in each of the two crisp clusters. Objects have partial memberships (or posterior probabilities) in  $U_2$ . For example, the first object is more similar to class 1 ( $u_{2,11}=0.7$ ) than to class 2 ( $u_{2,21}=0.3$ ). Because fuzzy partitions may indicate partial memberships in several clusters, they often provide valuable information about *twixters* (in-between objects) that is not available in a crisp partition. The most visible twixters in  $U_2$  are objects 3 and 4. When we want crisp labels from soft partitions, the usual method of "hardening" is to simply replace the maximum entry in each column of  $U$  by a 1, and place 0's in the remaining  $(c-1)$  columns (this is just Bayes rule when  $U$  is a probabilistic partition). We denote the *hardening* of  $U$  by  $H(U)$ . The last column in Table 1 has an example for the hardening of  $U_2$ . If we compare  $H(U_2)$  to  $U_1$ , we see that there is one *mismatch* (object 3). This is not necessarily an *error* – just a mismatch (it *would* be an error if  $U_1$  contained ground truth labels – i.e., labels known to be right).

**Table 1: Crisp, Soft and Hardened 2-partitions of  $n$  objects**

Crisp	Soft	Hardened $U_2$
$U_1 = \begin{pmatrix} 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 \end{pmatrix}$	$U_2 = \begin{pmatrix} 0.7 & 0.2 & 0.45 & 0.4 & 0.3 & 0.9 \\ 0.3 & 0.8 & 0.55 & 0.6 & 0.7 & 0.1 \end{pmatrix}$	$H(U_2) = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 & 1 & 0 \end{pmatrix}$

**2.2 Distinguished Features.** We want to cluster a sample of  $n$  objects  $O_n$  from the VL set  $O_N$ , or equivalently, we want clusters in a submatrix  $D_n$  of  $D_N$ . The progressive sampling scheme we have in mind is an adaptation of the object data scheme in Hathaway and Bezdek (2005). To use this approach we interpret the relational data as object data by regarding the  $i^{\text{th}}$  column  $\mathbf{D}^i$  of  $D_N$  (equivalently the  $i^{\text{th}}$  row, since  $D_N$  is symmetric) as a feature vector for object  $o_i$ .  $D_N = [\mathbf{D}^1 \mathbf{D}^2 \dots \mathbf{D}^N]$ . Any road map that has a table of distances between pairs of cities on the map is a relation of this type. This interpretation of  $\mathbf{D}^i$  is analogous to giving the location of city  $i$  (= object  $o_i$ ) by specifying - instead of, say, its rectangular coordinates - its distance to  $N-1$  other cities (i.e., the  $N-1$  distances between city  $i$  and the other cities on the map are its "features").

Which of these  $N$  "feature vectors" should we use? We believe it is reasonable to pick *distinguished features* (DFs) that are *very different* from each other. We choose  $h$ , the number of distinguished features to be selected, and  $H$ , which restricts the distinguished features to come only from rows 1 through  $H$  of  $D$ . This restriction allows the distinguished features selection portion of eNERF to be performed with only the top  $H \times H$  portion of  $D$ , which may be loadable when  $D$  is not.

**Algorithm DF (Select  $h$  distinguished features from  $H$  rows of  $D_N$ )**

**Choose:**  $h = \#$  of distinguished features to select  
 $H = \#$  of candidate rows for the  $h$  distinguished features,  $h \leq H$

**Input:** An  $H \times H$  dissimilarity matrix  $D_H$

**(DF1)** Define  $m_1 = 1$ . Initialize the search array

$$\boldsymbol{\delta}^1 = [\delta_1^1, \delta_2^1, \dots, \delta_H^1]^T = [d_{11}, \dots, d_{1H}]^T.$$

**(DF2)** Define  $m_2 = j$  where  $\delta_j^1 \geq \delta_k^1$  for  $1 \leq k \leq H$ .

**(DF3)** Define next search array  $\boldsymbol{\delta}^2 = [\delta_1^2, \delta_2^2, \dots, \delta_H^2]^T = [\min\{\delta_1^1, d_{m_2,1}\}, \dots, \min\{\delta_H^1, d_{m_2,H}\}]^T$

**(DF4)** Define  $m_3 = i$  where  $\delta_i^2 \geq \delta_k^2$  for  $1 \leq k \leq H$ .

**(DF5)** After  $j$  steps, use  $\boldsymbol{\delta}^j = [\delta_1^j, \delta_2^j, \dots, \delta_H^j]^T = [\min\{\delta_1^{j-1}, d_{m_j,1}\}, \dots, \min\{\delta_H^{j-1}, d_{m_j,H}\}]^T$  to choose the  $j+1^{\text{st}}$  feature as that row among the remaining candidates whose index points to the maximum element of  $\boldsymbol{\delta}^j$ .

If there is not a unique minimizing argument at some stage, ties can be broken by any rule. Why use the term "distinguished features"? Well, algorithm **DF** is a *feature selection* algorithm, but not in the usual pattern recognition sense of the term - i.e., we are not selecting good features for clustering or classifier design - rather, we are selecting good features for progressive sampling. It is our hope, of course, that these features will lead us to good clusters in  $D$ , but the quality of the features for

clustering does not determine their selection. However, we can relate algorithm **DF** to clusters in  $D$  in a very specific way.

Algorithm **DF** is justified in terms of sampling potential clusters by relating it to Dunn's index of separation (Dunn, 1976) which characterizes sets of clusters that are compact and well separated by a geometric criterion. The relevant result is

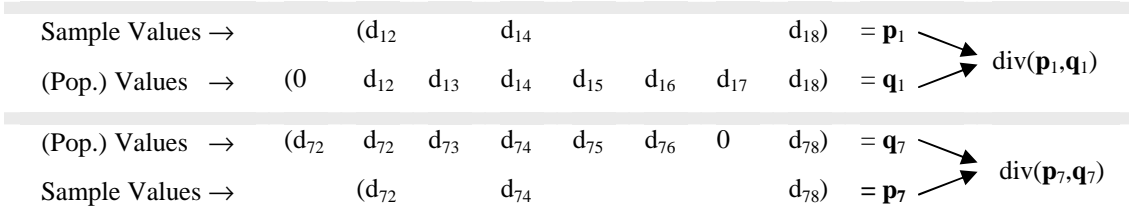
**Proposition DF.** If  $O_H$  has  $c$  compact and separated clusters, then the first  $c$  distinguished features chosen by algorithm **DF** will consist of one row corresponding to an object from each of the  $c$  clusters.

**Proof.** Given in Hathaway et al. (2005)

**2.3 Progressive sampling.** The criterion for acceptability of  $O_n$  is based on comparing the distribution of the distinguished features found by algorithm **DF** in the columns corresponding to  $O_n$  and  $O_N$  using the divergence test statistic. The scheme is best illustrated by a small example, shown in Figure 2, where we depict an  $8 \times 8$  dissimilarity matrix  $D_8$  enclosed in the heavy square, and 2 (arbitrarily chosen) distinguished features,  $m_1 = \text{feature 1}$  and  $m_2 = \text{feature 7}$ . Imagine now that the current candidate sample is  $O_3 = \{o_2, o_4, o_8\}$ . In essence, we will accept  $O_3$  as a representative sample of  $O_8$  if for *each* distinguished feature  $m_k$ , the distribution of the  $O_3$  feature values (corresponding to the shaded cells in row  $m_k$ ) closely approximates the distribution of the corresponding feature values for the full sample (corresponding to all entries in row  $m_k$ ).

More specifically, we accept  $O_3$  if the histogram of  $\{d_{12}, d_{14}, d_{18}\}$  is *close enough* to that of  $\{0, d_{12}, d_{13}, d_{14}, d_{15}, d_{16}, d_{17}, d_{18}\}$  and the histogram of  $\{d_{72}, d_{74}, d_{78}\}$  closely approximates that of  $\{d_{71}, d_{72}, d_{73}, d_{74}, d_{75}, d_{76}, 0, d_{78}\}$ . The approximation is "close enough" when the divergence test statistic is in the left tail of the appropriate chi-square distribution for each distinguished feature. In the bottom portion of Figure 2, we have extracted the relevant values needed for the divergence tests and shown them as vectors  $\mathbf{p}$  (sample values) and  $\mathbf{q}$  ("population" values). We will accept the sample if, and only if,  $\text{div}(\mathbf{p}_1, \mathbf{q}_1)$  AND  $\text{div}(\mathbf{p}_7, \mathbf{q}_7)$  are both less than or equal to  $F^{-1}(1-\epsilon)$ , where  $F$  is the *cumulative distribution function* (cdf) for the chi-square distribution with  $b-1$  degrees of freedom ( $b$  and  $\epsilon$  are discussed below). The use of a histogram-based test for acceptability requires selection of histogram bin intervals. Let  $b$  denote the desired number of histogram bins. The histogram interval widths are based only on the distinguished feature values of the initial sample, and the widths vary, as necessary, so that each histogram bin captures (as nearly as possible) the same number of initial sample observations. For notational simplicity we drop the second level of subscripts and let  $d_{(1)} \leq d_{(2)} \leq \dots \leq d_{(n)}$  denote the order statistics for the values of distinguished feature  $m_k$  from the columns of  $D$  corresponding to an initial sample  $O_n = \{o_{i_1}, o_{i_2}, \dots, o_{i_n}\}$ .

	o <sub>2</sub>		o <sub>4</sub>		o <sub>8</sub>			
DF m <sub>1</sub> = 1 →	0	<b>d<sub>12</sub></b>	d <sub>13</sub>	<b>d<sub>14</sub></b>	d <sub>15</sub>	d <sub>16</sub>	d <sub>17</sub>	<b>d<sub>18</sub></b>
	d <sub>21</sub>	0	d <sub>23</sub>	d <sub>24</sub>	d <sub>25</sub>	d <sub>26</sub>	d <sub>27</sub>	d <sub>28</sub>
	d <sub>31</sub>	d <sub>32</sub>	0	d <sub>34</sub>	d <sub>35</sub>	d <sub>36</sub>	d <sub>37</sub>	d <sub>38</sub>
	d <sub>41</sub>	d <sub>42</sub>	d <sub>43</sub>	0	d <sub>45</sub>	d <sub>46</sub>	d <sub>47</sub>	d <sub>48</sub>
	d <sub>51</sub>	d <sub>52</sub>	d <sub>53</sub>	d <sub>54</sub>	0	d <sub>56</sub>	d <sub>57</sub>	d <sub>58</sub>
	d <sub>61</sub>	d <sub>62</sub>	d <sub>63</sub>	d <sub>64</sub>	d <sub>65</sub>	0	d <sub>67</sub>	d <sub>68</sub>
DF m <sub>2</sub> = 7 →	d <sub>72</sub>	<b>d<sub>72</sub></b>	d <sub>73</sub>	<b>d<sub>74</sub></b>	d <sub>75</sub>	d <sub>76</sub>	0	<b>d<sub>78</sub></b>
	d <sub>81</sub>	d <sub>82</sub>	d <sub>83</sub>	d <sub>84</sub>	d <sub>85</sub>	d <sub>86</sub>	d <sub>87</sub>	0



**Figure 2: Progressive sampling termination testing using DFs 1 and 7 for sample objects 2, 4 and 8**

The  $b$  bin intervals of the histograms used to assess the suitability, according to distinguished feature  $m_k$ , of the original and all subsequent candidate samples are

$$\left[ 0, d_{\left(1 + \left\lceil \frac{n}{b} \right\rceil\right)} \right), \left[ d_{\left(1 + \left\lceil \frac{n}{b} \right\rceil\right)}, d_{\left(1 + \left\lceil \frac{2n}{b} \right\rceil\right)} \right), \right. \\ \left. \left[ d_{\left(1 + \left\lceil \frac{2n}{b} \right\rceil\right)}, d_{\left(1 + \left\lceil \frac{3n}{b} \right\rceil\right)} \right), \dots, \left[ d_{\left(1 + \left\lceil \frac{(b-1)n}{b} \right\rceil\right)}, \infty \right) \right)$$

where  $\lceil \cdot \rceil$  denotes the ceiling function. We refer to bins chosen this way as *equal content* bins, since the intervals divide the sample data exactly, or very nearly so, equally into the  $b$  bins.

Once the bin intervals are selected using the initial  $O_n$ , the same intervals are used - they are *not* redefined - for all subsequent candidates. For an example suppose that the initial sample is  $O_{45} = \{ o_{i_1}, o_{i_2}, \dots, o_{i_{45}} \}$  with the 45 corresponding sorted values  $d_{(1)} \leq d_{(2)} \leq \dots \leq d_{(45)}$  for distinguished feature  $m_k$ . Then for  $b = 4$ , the intervals defining the 4 bins are  $[0, d_{(13)})$ ,  $[d_{(13)}, d_{(24)})$ ,  $[d_{(24)}, d_{(35)})$ , and  $[d_{(35)}, \infty)$ . Assuming the 45 data values are all distinct, the (nearly equal) respective counts for the 4 intervals are 12, 11, 11, and 11.

There is one more issue concerning the sample data matrix that will be sent to LNERF for clustering. When

$O_n = \{ o_{i_1}, \dots, o_{i_n} \}$ , or equivalently, the corresponding columns of  $D_n$ , is accepted, then exactly what data are passed to LNERF? The sample used by LNERF consists of the set of all pair-wise dissimilarities of objects in  $\{ o_{i_1}, o_{i_2}, \dots, o_{i_n} \}$ , which is conveniently arrayed as an  $n \times n$  submatrix of the full sample matrix  $D_N$ . We denote this submatrix by  $D_n$ , where element  $d_{n_{jk}}$  is the pair-wise dissimilarity between sample objects  $o_{i_j}$  and  $o_{i_k}$ . For example, if the accepted sample is  $O_3 = \{ o_{i_1}, o_{i_2}, o_{i_3} \} = \{ o_2, o_4, o_8 \}$  as in Figure 2, then sample data matrix of dissimilarities that will be processed by LNERF is

$$D_3 = \begin{bmatrix} 0 & d_{24} & d_{28} \\ d_{42} & 0 & d_{48} \\ d_{82} & d_{84} & 0 \end{bmatrix}. \text{ Recall that we let } F \text{ denote the cdf}$$

for the chi-square distribution with  $b-1$  degrees of freedom. The sampling scheme is:

#### **Algorithm PS : Relational Progressive Sampling**

**Inputs:** An  $h \times N$  dissimilarity matrix  $D$ . The rows of  $D$  correspond to the  $h$  distinguished features  $\{ m_1, m_2, \dots, m_h \}$  selected by algorithm **DF** on  $D_H$

**Constraints:**  $D$  satisfies conditions (1)

**Choose:**  $b = \#$  of histogram intervals  
 $p =$  the initial sample percentage  
 $\Delta p =$  the incremental percentage

$\varepsilon_{PS}$  = termination criterion

(PS1) Randomly select (without replacement)  $n = \lceil (pN)/100 \rceil$  column indices  $I_n = \{c_1, \dots, c_n\}$  from  $I_N = \{1, 2, \dots, N\}$ .

(PS2) For  $k=1$  to  $h$ : define EC histogram bins for distinguished feature  $m_k$  with  $\{d_{m_k c_1}, d_{m_k c_2}, \dots, d_{m_k c_n}\}$ .

(PS3) For  $i=1$  to  $b$ ; for  $k=1$  to  $h$ :  
 Calculate  $N_i^k$ , the full set count for bin  $i$  and  $m_k$ , using row  $m_k$  of  $D$ .  
 Calculate  $n_i^k$ , the sample count for bin  $i$  and  $m_k$ , using  $\{d_{m_k c_1}, d_{m_k c_2}, \dots, d_{m_k c_n}\}$ .

(PS4) For  $k=1$  to  $h$ : calculate the divergence test criterion for distinguished feature  $m_k$

$$\text{div}_k = n \sum_{i=1}^b \left( \frac{N_i^k}{N} - \frac{n_i^k}{n} \right) \ln \left( \frac{n N_i^k}{N n_i^k} \right)$$

(PS5) **WHILE** ( $\text{div}_k > F^{-1}(1-\varepsilon)$  for at least one  $k \in \{1, 2, \dots, h\}$ )

$$\Delta n = \min\{N-n, (\Delta p N)/100\} : n = n + \Delta n$$

Randomly select  $\Delta D = (\Delta n)$  previously unselected columns of  $D$

$D_n = D_n + \Delta D$  % Adding  $\Delta n$  columns to  $D_n$  also adds  $\Delta n$  rows to  $D_n$

$$\text{Calculate } \text{div}_k = n \sum_{i=1}^b \left( \frac{N_i^k}{N} - \frac{n_i^k}{n} \right) \ln \left( \frac{n N_i^k}{N n_i^k} \right); k = 1, \dots, h$$

**Output :**  $n \times n$  (Sample) Dissimilarity matrix  $D_n$

**2.4 LNERF.** Clustering in  $D_n$  is done with the literal NERF algorithm from Hathaway and Bezdek (1994). In the following,  $\|\cdot\|$  is the Euclidean norm on  $\mathfrak{R}^n$ ,  $\mathbf{e}_k$

denotes the  $k^{\text{th}}$  unit vector in  $\mathfrak{R}^n$ ,  $M$  is the  $n \times n$  matrix with 0's on the main diagonal and 1's elsewhere ( $M = [\mathbf{1}] - I_n$ ), and  $M_{fcm}$  is the set of all fuzzy partition

matrices  $U \in \mathfrak{R}^{cn}$  which satisfy  $u_{ik} \in [0, 1]$ ,  $\sum_{i=1}^c u_{ik} = 1$  for

$k = 1, \dots, n$  and  $\sum_{k=1}^n u_{ik} > 0$  for  $i = 1, \dots, c$ . Don't confuse

$U^{(q)}$ , the  $q^{\text{th}}$  estimate of the  $c \times n$  matrix  $U$ , with our notation  $U^q$ , which denotes the  $q^{\text{th}}$  column of matrix  $U$ .

**Algorithm LNERF : Fuzzy clusters in dissimilarity matrix  $D_n$  (Hathaway and Bezdek, 1994)**

**Inputs:** An  $n \times n$  dissimilarity matrix  $D_n$

**Constraints:**  $D_n$  satisfies conditions (1) and  $\alpha \in \mathfrak{R}$ ,  $M = [\mathbf{1}] - I_n$

**Choose:**  $c = \#$  of clusters,  $2 \leq c < n$   
 $m =$  fuzzy weighting exponent,  $m > 1$   
 $\varepsilon_L =$  termination criterion  
 $\|U^{(q)} - U^{(q-1)}\| =$  termination norm  
 $Q_M =$  maximum number of iterations

**Initialize:**  $q = 0$ ;  $\beta = 0$ ;  $U^{(0)} \in M_{fcm}$   
 $U_{\text{difference}} = 2 * \varepsilon_L$

**WHILE** ( $U_{\text{difference}} > \varepsilon_L$  **AND**  $q < Q_M$ )

(LN1) For  $1 < I < c$ , calculate "mean" vector  $\mathbf{v}_i^{(r)}$

$$\mathbf{v}_i^{(q)} = ((u_{i1}^{(q)})^m, \dots, (u_{in}^{(q)})^m)^T / \sum_{j=1}^n ((u_{ij}^{(q)})^m) \quad ; \quad (2a)$$

(LN2) Calculate (cn) object to cluster "distances"

$$\delta_{ik} = ((D_n + \beta M) \mathbf{v}_i^{(q)})_k - (\mathbf{v}_i^{(q)})^T (D_n + \beta M) \mathbf{v}_i^{(q)} / 2 \quad (2b)$$

**IF**  $\delta_{ik} < 0$  for any  $i$  and  $k$ , **THEN** calculate

$$\Delta \beta = \max_{i,k} \left\{ -2\delta_{ik} / \|\mathbf{v}_i^{(q)} - \mathbf{e}_k\|^2 \right\} \quad ; \quad (2c)$$

$$\delta_{ik} \leftarrow \delta_{ik} + (\Delta \beta / 2) \cdot \|\mathbf{v}_i^{(q)} - \mathbf{e}_k\|^2 \quad ; \quad (2d)$$

$$\beta \leftarrow \beta + \Delta \beta \quad . \quad (2e)$$

(LN3) Update  $U^{(q)}$  to  $U^{(q+1)} \in M_{fcm}$  for all  $k=1, \dots, n$ :

**IF**  $\delta_{ik} > 0$  for any  $i = 1$  to  $c$  **THEN**

$$u_{ik}^{(q+1)} = 1 / \left[ \sum_{j=1}^c (\delta_{ik} / \delta_{jk})^{1/(m-1)} \right] \quad ; \quad (2f)$$

**ELSE**

$$u_{ik}^{(q+1)} = 0 \text{ for all } k \text{ with } \delta_{ik} > 0$$

$$u_{ik}^{(q+1)} \in [0, 1] \text{ such that } \sum_{j=1}^c u_{jk}^{(q+1)} = 1 \quad (2g)$$

$$q \leftarrow q + 1$$

$$U_{\text{difference}} = \|U^{(q)} - U^{(q-1)}\|$$

**Outputs :** Membership matrix  $U_{\text{LNERF}, n} \in M_{fcm}$ ;  
 "prototype" vectors  $\{\mathbf{v}_1, \dots, \mathbf{v}_c\} \subset \mathfrak{R}^n$

The main result in Hathaway and Bezdek (1996) is that the sequence of partition matrices produced by LFCM on an object data set  $X$  is identical to the sequence of partition matrices produced by LNERF on the corresponding relational matrix  $D(X)$  of pair-wise squared Euclidean distances derived from  $X$ , i.e.,

$$[d_{ij}(X)] = \left[ \|\mathbf{x}_i - \mathbf{x}_j\|^2 \right]. \text{ The "non-Euclidean" terminology}$$

indicates that LNERF is applicable to non-Euclidean data.

Continuation of the algorithm is achieved by  $\beta$ -spreading  $D$  to  $D_\beta = D_n + \beta M$  via equations (2b)-(2e) which, when triggered, add spread term  $\beta$  to the off diagonal elements of  $D$ . Addition of the term  $\beta M$  produces relational dissimilarity data that is very nearly Euclidean, while preserving most of the cluster structure of the original  $D$ . The  $\{\delta_{ik}\}$  shown in equations (2) are analogous to distances in LFCM, the object data dual of LNERF, but are not true distances in the relational data setting. Finally, the specific norm used in LNERF for termination is important only in that different norms may stop the algorithm at different elements in the iterate sequence.

**2.5 Extension.** The final component of eNERF is extension of LNERF results on  $O_n$  to the objects in  $(O_N - O_n)$ . We temporarily denote the LNERF clustering results by  $(U = U_{\text{LNERF}}, \mathbf{V})$  without subscripts, where  $U = [u_{ik}] \in M_{\text{fcn}}$  and  $\mathbf{V} = [\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_c] \in \mathfrak{R}^{nc}$ . The result of extension is an augmented  $c \times N$  matrix  $U_{\text{app}}$  obtained by appending  $(N - n)$  columns from xNERF to the  $c \times n$  matrix  $U$ .  $U_{\text{app}}$  is the approximation to  $U_{\text{lit}}$ , the partition of  $D_{\text{VL}}$  that we seek, but cannot compute. *Extended NERF* (xNERF) calculates  $U^j = (u_{1j}, \dots, u_{cj})^T$ , the membership column in the output matrix  $U_{\text{xNERF}}$  corresponding to  $o_j$ , for  $n+1 \leq j \leq N$ . Unlike the direct calculation used to extend FCM in the image and general object data cases, xNERF extension requires iteration to produce a label vector for each of the  $N-n$  unlabelled objects. This sounds worse, complexity-wise, than it is. Several vector computations are done just once. Subsequently, only simple scalar quantities are recalculated, so the iteration that produces each  $U^j$  is fairly inexpensive.

Calculating memberships for  $o_j$  begins by appending some of  $o_j$ 's corresponding relational data to the sample relational matrix  $D_n$  to obtain an  $(n+1) \times (n+1)$  augmented relational matrix. Then LNERF is applied to this augmented matrix, but the sample-based estimates of  $U$  and  $\mathbf{V}$  from LNERF are *fixed*.

Let  $\beta_n$  denote the final shifting value obtained from (2e) while clustering  $D_n$  using LNERF; and let  $D_{\beta_n}$  denote the  $\beta_n$  shifted version of  $D_n$ ,  $D_{\beta_n} = D_n + \beta_n M$ . Let  $D_N = [d_{ik}]$  denote the original  $N \times N$  full relation matrix. For convenience, we assume without loss of generality that the objects in  $O_N$  have been re-indexed so that the unlabeled objects are adjacent,  $O_N - O_n = \{o_{n+1}, o_{n+2}, \dots, o_N\}$ . To assign a fuzzy label vector to the  $j^{\text{th}}$  object in  $O_N - O_n$ , we first define some auxiliary quantities built from the values of row or column  $j$  of  $D_N$  and the outputs of LNERF on  $D_n$ . Let  $\mathbf{d}_j = (d_{1j}, \dots, d_{nj})^T$ ,  $\boldsymbol{\beta}_n = (\beta_n, \dots, \beta_n)^T$  and  $\mathbf{z}_j = (\mathbf{d}_j + \boldsymbol{\beta}_n)$  be  $n$ -vectors. Next, define the  $(n+1) \times (n+1)$  augmented matrix  $D_j$  as

$$D_j = \begin{bmatrix} D_{\beta_n} & \mathbf{z}_j \\ \mathbf{z}_j^T & 0 \end{bmatrix} \quad (11)$$

For  $i = 1, \dots, c$ , let

$$a_i = \sum_{k=1}^n u_{ik}^m \text{ using the terminal } U \text{ from (2g, h)}, \quad (3a)$$

$$b_i = \mathbf{v}_i^T D_{\beta_n} \mathbf{v}_i \text{ using the terminal } \mathbf{v}_i \text{'s from (2a)}, \quad (3b)$$

$$c_i = \mathbf{z}_j^T \mathbf{v}_i = \langle \mathbf{z}_j, \mathbf{v}_i \rangle \quad (3c)$$

These three quantities are used by xNERF to iteratively estimate two sets of unknowns for the object  $o_j \in O_N - O_n$ . The unknowns that will be estimated are:  $c$  values  $\{v_{ij} : 1 \leq i \leq c\}$ , which correspond to the  $n+1^{\text{st}}$  "components" of new prototype vectors gotten by applying LNERF to a distance matrix of size  $(n+1) \times (n+1)$ ; and a  $c \times 1$  label vector  $\mathbf{U}^j \in N_{\text{fc}}$ . Next we append the unknown components of the prototypes we seek to the input prototypes from LNERF:

$$\mathbf{v}_{ij} = [\mathbf{v}_i^T, v_{ij}]^T ; 1 \leq i \leq c \quad (4)$$

Let  $\mathbf{1} = (1, 1, \dots, 1)^T \in \mathfrak{R}^n$  and define  $M_j$  as

$$M_j = \begin{bmatrix} \mathbf{0}_{n \times n} & \mathbf{1} \\ \mathbf{1}^T & 0 \end{bmatrix} \in \mathfrak{R}^{(n+1) \times (n+1)} \quad (5)$$

We use  $\mathbf{1}^T \mathbf{v}_i = 1$ , due to normalization of the sample prototype vectors during LNERF processing of  $D_n$ , to simplify our description of xNERF. Let  $\tau \in \mathfrak{R}$  be any real number, and calculate

$$\begin{aligned} \left( (D_j + \tau M_j) \mathbf{v}_{ij} \right)_{n+1} &= \left( \begin{bmatrix} D_{\beta_n} & \mathbf{z}_j + \tau \mathbf{1} \\ \mathbf{z}_j^T + \tau \mathbf{1}^T & 0 \end{bmatrix} \begin{bmatrix} \mathbf{v}_i \\ v_{ij} \end{bmatrix} \right)_{n+1} ; \quad (6a) \\ &= \mathbf{z}_j^T \mathbf{v}_i + \tau \mathbf{1}^T \mathbf{v}_i = c_i + \tau \end{aligned}$$

Also,

$$\begin{aligned} (\mathbf{v}_{ij})^T (D_j + \tau M_j) (\mathbf{v}_{ij}) &= \begin{bmatrix} \mathbf{v}_i^T & v_{ij} \end{bmatrix} \begin{bmatrix} D_{\beta_n} & \mathbf{z}_j + \tau \mathbf{1} \\ \mathbf{z}_j^T + \tau \mathbf{1}^T & 0 \end{bmatrix} \begin{bmatrix} \mathbf{v}_i \\ v_{ij} \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{v}_i^T & v_{ij} \end{bmatrix} \begin{bmatrix} D_{\beta_n} \mathbf{v}_i + v_{ij} (\mathbf{z}_j + \tau \mathbf{1}) \\ c_i + \tau \end{bmatrix} \\ &= \mathbf{v}_i^T D_{\beta_n} \mathbf{v}_i + v_{ij} (\mathbf{v}_i^T \mathbf{z}_j + \tau \mathbf{v}_i^T \mathbf{1}) + v_{ij} (c_i + \tau) \\ &= b_i + 2v_{ij} (c_i + \tau) \quad (6b) \end{aligned}$$

Using (6) we can simplify the quantity

$$\begin{aligned} \left( (D_j + \tau M_j) \mathbf{v}_{ij} \right)_{n+1} - \frac{1}{2} (\mathbf{v}_{ij})^T (D_j + \tau M_j) \mathbf{v}_{ij} &= \\ c_i + \tau - \frac{1}{2} (b_i + 2v_{ij} (c_i + \tau)) &= (c_i + \tau)(1 - v_{ij}) - \frac{b_i}{2} \quad (7) \end{aligned}$$

Now we are ready to state the extension procedure.

**Algorithm xNERF (Extension of LNERF to label object  $o_j \in O_N - O_n : j = n+1, \dots, N$ )**

**Inputs:** From algorithm (PS):  $D_n$

From algorithm (**LNERF**):

$$\beta_n; D_{\beta_n} = D_n + \beta_n(M);$$

$$U = U_{\text{LNERF}} \in M_{fc_n}; \mathbf{V}_{c \times n}$$

$m =$  fuzzy weighting exponent,  $m > 1$

**Choose:**  $\epsilon_x =$  termination criterion  
 $Q_M =$  maximum number of iterations  
 $\|U^{(q)} - U^{(q-1)}\| =$  termination norm

**Calculate:**  $a_i$  and  $b_i$  from (12) for  $i = 1$  to  $c$ .

**FOR**  $j = n+1$  to  $N$

(**xN 1**) **Initialize:**  $q = 0; \beta = 0; U_{\text{difference}} = 2^* \epsilon_x;$

$$U^{(0)} = (1/c, \dots, 1/c)^T \in N_{fc};$$

Calculate  $c_i$  from (3c) for  $i = 1$  to  $c$ .

(**xN 2**) **WHILE** ( $U_{\text{difference}} > \epsilon_x$  **AND**  $q < Q_M$ )

Calculate new prototypes  $\mathbf{v}_{ij}^{(q)} \in \mathfrak{R}^{n+1}$  for  $i = 1$  to  $c$ :

$$\mathbf{v}_{ij}^{(q)} = \left[ \mathbf{v}_i^T, \frac{k_{ij}}{(a_i + k_{ij})} \right]^T, \quad \text{where} \quad k_{ij} = \left( u_{ij}^{(q)} \right)^m$$

Calculate new "distances" (when positivity is violated) from  $o_j$  to the  $c$  clusters for  $i = 1$  to  $c$ :

**IF**  $\delta_{ij} < 0$  for any  $i \in \{1, \dots, c\}$ , **THEN**

$$\Delta\beta = -\min_i \{\delta_{ij}\}$$

$$\delta_{ij} \leftarrow \delta_{ij} + \Delta\beta \text{ for } i = 1 \text{ to } c$$

$$\beta \leftarrow \beta + \Delta\beta$$

**ENDIF**

Calculate  $\{u_{ij}^{(q+1)}\}$  for  $o_j$  for  $i = 1$  to  $c$ :

**IF**  $\delta_{ij} > 0$ ,  $i = 1$  to  $c$  **THEN**

$$u_{ij}^{(q+1)} = \left( \frac{\sum_{h=1}^c \delta_{ij}}{\delta_{ij}} \right)^{-1}$$

**ELSE**

$$u_{ij}^{(q+1)} = 0 \text{ for all } k \text{ with } \delta_{ik} > 0, \text{ and}$$

$$u_{ij}^{(q+1)} \in [0, 1] \text{ such that } \sum_{j=1}^c u_{ij}^{(q+1)} = 1$$

**ENDIF**

Update  $q \leftarrow q + 1$

$$\text{Calculate } U_{\text{difference}} = \|U^{(q)} - U^{(q-1)}\|$$

**END WHILE**

$$U^j = (u_{1j}^{(q)}, \dots, u_{cj}^{(q)})^T$$

**NEXT**  $j$

**Output :** Matrix  $U_{\text{xNERF}} = [U^j, \dots, U^N] \in M_{fc(N-n)}$

**Remarks.** (i) it is *never* necessary to load the entire matrix  $D_N = D_{VL}$ . Finding the  $h$  DFs that monitor sampling uses an  $H \times H$  submatrix of  $D_N$ . (ii) progressive sampling operates on an  $h \times N$  portion of  $D_N$ , which is just the  $h$  rows of  $D_N$  corresponding to the  $h$  DFs. (iii) LNERF clustering operates on the  $n \times n$  matrix  $D_n$ , where  $n$  is the size of the accepted sample. (iv) xNERF accesses elements in an  $n \times (N-n)$  portion of  $D_N$ .

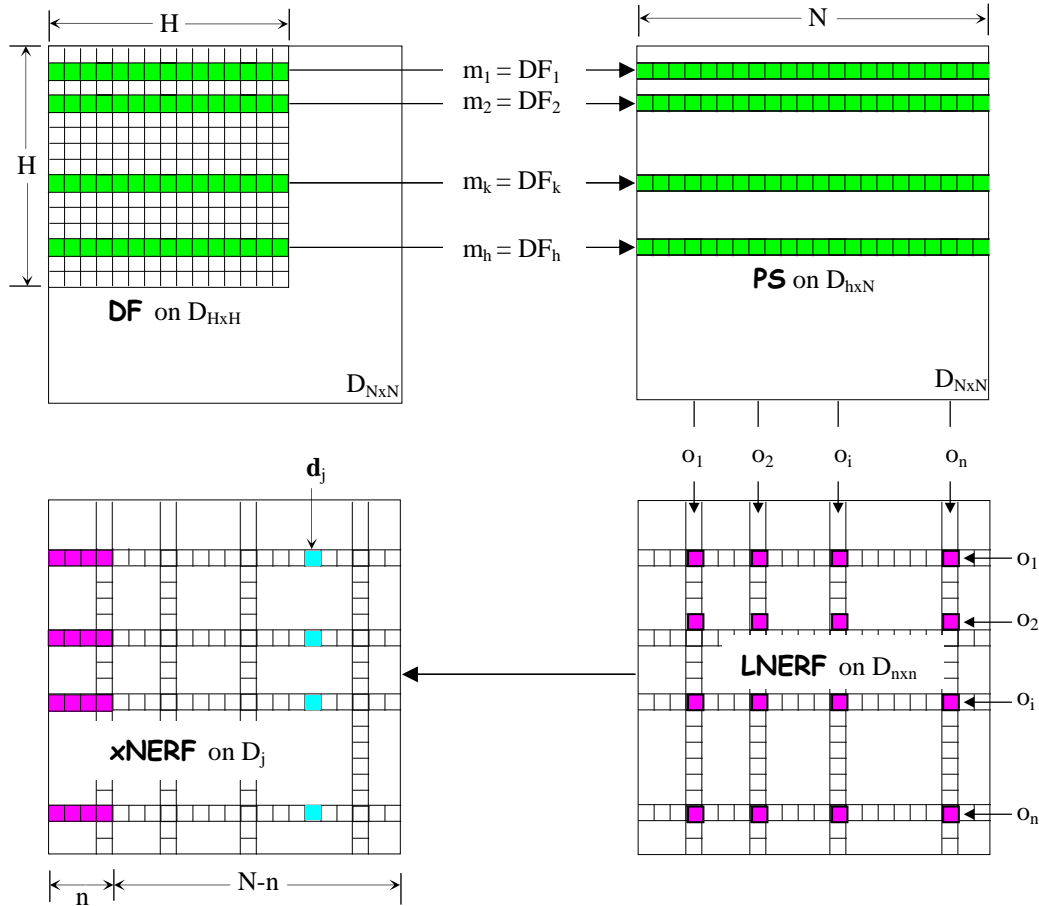
Figure 3 summarizes the **eNERF** approach to clustering in VL dissimilarity data, which consists of the sequential application of four algorithms: **DF**, **PS**, **LNERF** and **xNERF**. The final operation in this sequence is to form a  $c$ -partition of  $O_N$  (if this is desired), which is done by concatenating  $U_{\text{xNERF}, N-n}$  with  $U_{\text{LNERF}, n}$ , thereby creating  $U_{\text{app}} = [U_{\text{LNERF}, n} | U_{\text{xNERF}, N-n}]_{c \times N}$ , and the approximation to literal clusters in  $D_N$ , which cannot be computed when  $D_N$  is unloadable. The shaded cells in the upper left panel in Figure 3 depict the  $H \times H$  matrix from which the  $h$  distinguished features  $\{m_k\}$  are chosen by algorithm DF.

The shaded rows in the upper right panel are the full rows of  $D_N$  corresponding to the  $h$  rows in  $D_H$ . This is the input matrix to our progressive sampling scheme. The shaded cells in the lower right panel are the entries of  $D_n$ , the sample matrix processed by LNERF. And finally, the lower left panel depicts the reindexed version of  $D_N$ . The blocks of shaded cells in the first  $n$  columns are the fixed values of  $D_n$ , while the  $n$  shaded cells in column  $j$ ,  $n+1 \leq j \leq N$ , are the components of the vector  $\mathbf{d}_j$ . The matrix  $D_j$  which is alluded to in this view is built from the shaded cells in this view and the terminal  $\beta$ -spread value  $\beta_n$  from LNERF.

### 3 Numerical Examples

The termination norms for the two examples are: for LNERF, the sup norm for matrices, regarding them as vectors in  $\mathfrak{R}^{cn}$ ; and for xNERF, the sup norm on  $\mathfrak{R}^c$ . The first example gives a visual display of the quality of an extended partition using a  $194 \times 194$  matrix consisting of pair-wise similarities from a set of human gene products. The second example demonstrates eNERF on a problem that is too big for the PC used in the calculations, which consisted of  $40,000 \times 40,000$  dissimilarities. Storage of this matrix can be cut in half by storing just the upper triangular part of the symmetric input matrix, but the data are still too large to be handled directly with LNERF.

**Example 1.** We demonstrate eNERF by clustering a subset of GDP194<sub>4,30.04</sub> with LNERF, extending the results to the remaining objects with xNERF, and then comparing  $U_{\text{app}} = [U_{\text{LNERF}, n} | U_{\text{xNERF}}]_{c \times N}$  to  $U_{\text{LNERF}, N}$ .



**Figure 3: Architecture of eNERF clustering in VL dissimilarity data**

GDP194<sub>4,30,04</sub> is a 194 x 194 matrix whose entries are similarities between pairs of gene products developed on sets of linguistic descriptors of each protein in the set; see Pal et al. (2005) for a complete description of the construction and interpretation of this data. The specific matrix we use here was called LOS4. The subscript indicates the date that the data was originally collected. Since the original similarity values of the data are all in [0,1], we obtained dissimilarities by subtracting each similarity from 1.

There is evidence (from human experts and other clustering algorithms) to believe that the data consists of 3 main clusters, which group them into three protein families {myotubularin, receptor precursor, collagen alpha chain}. The data were ordered so that the first 21 columns (or rows) corresponded to the first cluster, as did the next 87, and the final 86. Hardening an LNERF fuzzy 3-partition of the data produces these same clusters of sizes 21, 87, and 86.

Relational data and clusters in it can be displayed as a grayscale image. The input dissimilarity matrix for GDP194<sub>4,30,04</sub>, in the original order (clustered by human experts), is shown in Figure 4(a). The three dark blocks along the main diagonal correspond to the three main clusters.

The purpose of this example is to demonstrate the visual quality of extended partitions using different percentages of data in the sample. To do this we need to randomly sample a percentage of data from the matrix  $D$  corresponding to Figure 4(a). One way to randomly sample from the columns is to randomly permute the rows and columns of  $D$  by the same permutation of  $1, 2, \dots, 194$ . After permutation, a random sample can be easily obtained by choosing, in order, columns from the scrambled version of  $D$ . This approach allows us to better "see" the random sample. Using MATLAB routine randperm, seeded with value 824567, we obtained the scrambled form of the dissimilarity data shown in Figure 4(b), which we denote here as  $D_{\text{scram}}$ .

Visual information about clusters in a dissimilarity image (Figure 4(a)) is also available by viewing a transformed version of a fuzzy partition found by clustering the data. Figure 5 demonstrates this with images based on extended fuzzy membership matrices computed from the first (a) 10% of  $D_{\text{scram}}$ , (b) 25% of  $D_{\text{scram}}$ , (c) 75% of  $D_{\text{scram}}$ , and (d) 100% of  $D_{\text{scram}}$  (LNERF on  $D_{\text{scram}}$ ). To appreciate the extended partition images, note that these percentages are based on the ratio of sample size  $n$  to full sample size  $N=194$ . So the 10% case corresponds to a sample dissimilarity matrix  $D_{19}$  consisting of the  $19 \times 19$  leading principal submatrix of  $D_{\text{scram}}$ .

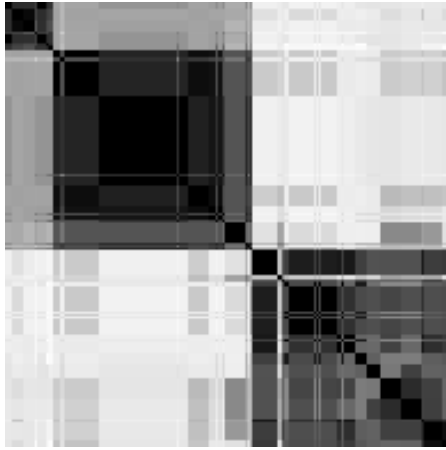


Figure 4a: Input dissimilarity matrix  $D_{194}$

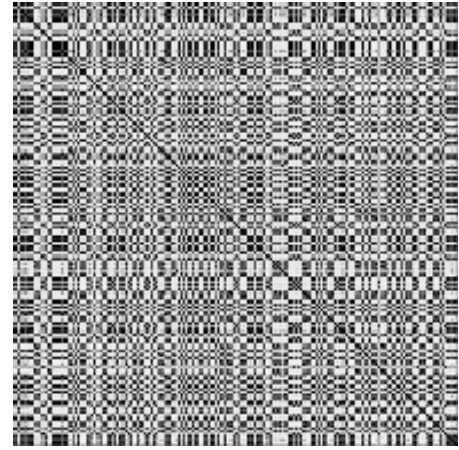


Figure 4b:  $D_{\text{scram}}$  reordering of  $D_{194}$

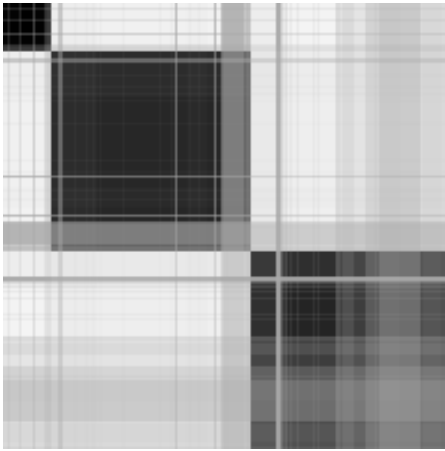


Figure 5a: 10% sample (n=19)

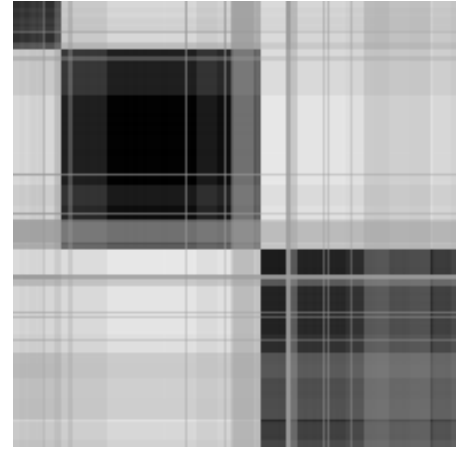


Figure 5c: 75% sample (n=144)

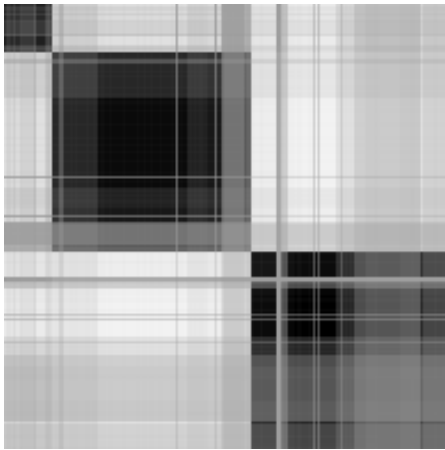


Figure 5b: 25% sample (n=48)

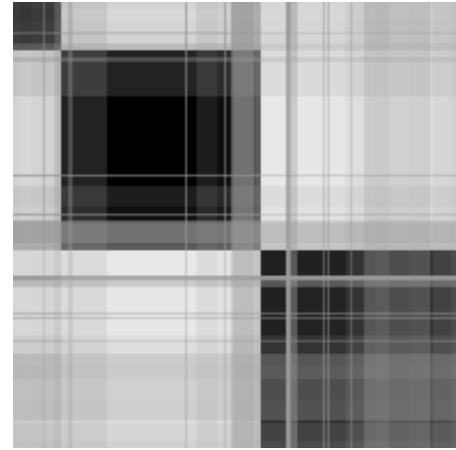


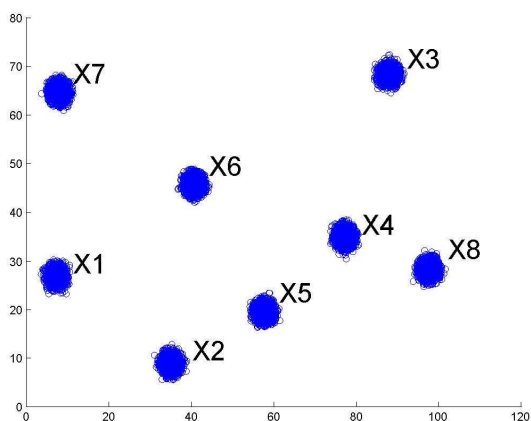
Figure 5d: 100% (n=194:  $U_{\text{LNERF},N}$ )

The percentage of elements (not columns) of  $D_{\text{scram}}$  actually used by LNERF in the clustering phase is about 1 percent. Likewise, using 75% of the 194 objects in our scheme is equivalent to using  $9/16 \approx 56\%$  of the elements of  $D_{\text{scram}}$ . The images in Figure 5 were produced from fuzzy partitions of  $D_{\text{scram}}$  in the following way. The columns of the fuzzy partition  $U_{\text{scram}}$  of  $D_{\text{scram}}$  are reordered (unscrambled) to get  $U$ , corresponding to the same object ordering as in Figure 4(a). Then the corresponding image matrix  $I(U) = J - (U^T U / \max(U^T U))$

where  $J$  denotes the  $194 \times 194$  matrix of 1's and  $\max(U^T U)$  denotes the largest element in  $U^T U$ . Figures 5(a, d) show that the extended partition based on only 10% of  $D_{194}$  is a close (visual) approximation to the result obtained by applying LNERF to  $D_{194}$  itself. The extended partition images for successively larger samples of  $D_{194}$  in views (a)-(c) are increasingly better approximations to the literal partition seen in view (d). Note, too, the clear similarity between all of the partition-based images in Figure 5 and the dissimilarity-based

image of Figure 4 (a). And finally, we observe that when analyzing clustering outputs with images such as these, there is no need to harden fuzzy partitions – these images are built directly with the fuzzy memberships produced by LNERF and xNERF.

**Example 2.** This example demonstrates the feasibility property of eNERF by clustering a  $40,000 \times 40,000$  relational data matrix derived as pair-wise squared Euclidean distances from a set of 40,000 2-dimensional points which form 8 very well separated, compact clusters. This data set was first used for testing a visual display technique in Huband et al. (2005), and is represented by a scatter plot in Figure 6.



**Figure 6. Scatter plot of 8-cluster, 40000-point data set in  $\mathcal{R}^2$  from Huband et al. (2005)**

The 8-cluster data set can be clustered by almost any object data clustering algorithm without difficulty on the PC used for this experiment. In particular, we can cluster this data with LFCM in  $\mathcal{R}^2$ , so the full data fuzzy partition  $U_{lit}$  is obtainable, and by the duality theory, identical to the one obtained by running LNERF on the corresponding distance matrix. But for  $N=40000$ , we are not able to calculate, load and process the full distance data matrix  $D_N$ . In other words, if we had *only* the relational data, this *would* represent a VERY LARGE clustering problem relative to the computing environment available. In fact, an  $800 \times 40000$  slice (1/50) of  $D_N$  requires 244 MB of storage as a MATLAB \*.MAT file.

To cluster  $D$  using eNERF we first performed progressive sampling with the following parameter values: divergence acceptance threshold  $\epsilon_{ps} = 0.80$ ; number of DF candidates  $H = 10$ ; number of DFs  $h = 1$ ; number of histogram bins  $b = 10$ ; initial sample percentage  $p = 1$  (% of  $N = 40000$ ) = 400; and incremental sample percentage  $\Delta p = 1$  (% of  $N = 40000$ ) = 400. The choice  $h = 1$  means that, for any choice of  $H$ , the only DF chosen is the first feature (row 1). The PS scheme terminated after one increment, the divergence accepting a representative sample of size  $n = 800$ .

For LNERF, we used  $m = 2$ ,  $\epsilon_L = 0.00001$ . Initializing with the correct hard 8-partition of the data, the  $800 \times 800$  submatrix  $D_n$  was clustered after 3 iterations. The xNERF phase required a  $800 \times 40000$  slice of  $D$  which had storage requirements of 244 MB. To avoid "out of memory" errors with MATLAB, the processing was broken up by calling the extension routine 49 times, each time supplying it with  $D_n$  and an additional  $800 \times 800$  sub block of  $D_N$ . This chunk was used to extend the partition for another 800 objects. The stopping criterion for the extension iteration used  $\epsilon_x = 0.001$ , and the final extended result  $U_{app}$  satisfied  $\|U_{lit} - U_{app}\|_F = 0.2548$ , which is quite small since these matrices are  $8 \times 40000$ . This was certainly an easy problem, in terms of how well separated the clusters actually are, but the point here was to demonstrate the feasibility property of eNERF, and this example does that.

#### 4. Discussion

We have shown how to extend NERF clustering to arbitrarily large dissimilarity data. Several opportunities for future work immediately come to mind. An efficient implementation of eNERF, particularly for xNERF, needs to be made to find out to what degree, and for what problems, eNERF can provide acceleration to LNERF. A theoretical analysis of the convergence properties of xNERF would be of value and would likely be possible using the alternating optimization framework of Bezdek and Hathaway (2003). One of the most interesting issues concerns the selection of the optimal distinguished features, those that best provide samples that correlate well with accurate extended partitions. Is an alternative, non-iterative, extension possible? We did very limited experimentation using a direct FCM-based extension scheme, which proved to be faster, but much less accurate. Finally, the interesting discovery regarding the similarity between partition and relational based images in Example 1 suggests a potentially cheaper (since  $U$  is smaller than  $D$ ) approach than those in Huband et al. (2005) for visual displays to assess cluster tendency and validity.

#### 5 References

- Bezdek, J.C., Keller, J.M., Krishnapuram, R. and Pal, N.R., (1999): *Fuzzy models and algorithms for pattern recognition and image processing*. Springer, NY.
- Bezdek, J. C. and Hathaway, R.J. (2003): Convergence of alternating optimization. *Neural, Parallel and Scientific Computations*, **11**, 351-368.
- Bradley, P., Fayyad, U. and Reina, C. (1998): Scaling clustering algorithms to large databases. *Proc. 4<sup>th</sup> Int'l. Conf. Knowledge Discovery and Data Mining*, 9-15, AAAI Press, Menlo Park, CA,.
- Dunn, J. C. (1976): Indices of partition fuzziness and the detection of clusters in large data sets, in *Fuzzy Automata and Decision Processes*, M. M. Gupta (ed), Elsevier, NY.

- Fayyad, U. and Smyth, P. (1996): From massive data sets to science catalogs: applications and challenges. *Proc. Workshop on Massive Data Sets*, J. Kettinger and D. Pregibon (eds), National Research Council.
- Ganti, V., Ramakrishnan, R., Gehrke, J., Powell, A. L. and French, J. C. (1999): Clustering large data sets in arbitrary metric spaces. *Proc. 15<sup>th</sup> Int'l. Conf. on Data Engineering*, 502-511, IEEE CS Press, Los Alamitos, CA.
- Hathaway, R. J. and Bezdek, J. C. (1994): NERF c-means: non-Euclidean relational fuzzy clustering. *Patt. Recog.*, **27(3)**, 429-437.
- Hathaway, R.J. and Bezdek, J.C. (2005). Approximate clustering in very large data sets. In press, *Comp. Statistics and Data Analysis*.
- Hathaway R. J., Bezdek, J. C., Huband, J. M., Leckie, C. and Kotagiri, R. (2005): Approximate clustering in very large relational data, in review, *Jo. Intell. Syst.*
- Huband, J., Bezdek, J. C. and Hathaway, R J. (2005): bigVAT: visual assessment of cluster tendency for large data sets. *Patt. Recog.*, **38**, 1875-1886.
- Huber, P., (1996): Massive data workshop: The morning after. *Massive Data Sets*, 169-184, National Academy Press.
- Ng, R. T. and Han, J. (1994): Efficient and effective clustering methods for spatial data mining. *Proc. 20th Int'l. Conf. On Very Large Databases*, 144-155, Morgan Kaufman, San Francisco.
- Pal, N.R. and Bezdek, J.C. (2002): Complexity reduction for "large image" processing. *IEEE Trans. on Systems, Man and Cybernetics*, **B(32)**, 598-611.
- Pal, N. R, Keller, J. M., Mitchell, J.A., Popescu, M., Huband, J. M. and Bezdek, J.C. (2005): Gene ontology-based knowledge discovery through fuzzy cluster analysis, in press, *Neural, Parallel and Scientific Computing*.
- Taskar, B., Segal, E. and Koller, D. (2001). Probabilistic clustering in relational data. *17<sup>th</sup> International Joint Conference on Artificial Intelligence*, 870-876, Seattle, USA.