

# Improved visual clustering of large multi-dimensional data sets

Eduardo Tejada

Institut for Visualization and Interactive Systems  
University of Stuttgart  
eduardo.tejada@vis.uni-stuttgart.de

Rosane Minghim

Institut of Mathematics and Computer Science  
University of Sao Paulo  
rminghim@icmc.usp.br

## Abstract

*Lowering computational cost of data analysis and visualization techniques is an essential step towards including the user in the visualization. In this paper we present an improved algorithm for visual clustering of large multi-dimensional data sets. The original algorithm is an approach that deals efficiently with multi-dimensionality using various projections of the data in order to perform multi-space clustering, pruning outliers through direct user interaction. The algorithm presented here, named HC-Enhanced (for Human-Computer enhanced), adds a scalability level to the approach without reducing clustering quality. Additionally, an algorithm to improve clusters is added to the approach. A number of test cases is presented with good results.*

## 1 Introduction

Clustering large multi-dimensional data sets presents two major problems besides generating a good clustering: scalability and capacity for dealing with multi-dimensionality. Scalability usually means linear complexity ( $O(n)$ ) or better ( $O(n \log n)$ ,  $O(\log n)$ , etc.). For highly interactive systems, such as visual clustering techniques [1, 11, 16], scalability is critical. Algorithms with linear complexity but high constant ( $O(kn)$ ,  $k$  constant) are not suitable for such systems: interactivity is a key issue that must be included in visual mining techniques [20], thus the response time must be as low as possible.

Treating multi-dimensionality is also a very difficult task. The research done for tackling this problem are based in three approaches: subspace clustering, co-clustering [4], and feature selection [19]. These approaches are focused on solving the problem known as the “dimensionality curse”, which is the incapacity for generating significative structures (patterns or models) from high dimensional data. For clustering algorithms in most cases this means more than 15 dimensions [4, 5].

Subspace clustering refers to approaches that apply dimensionality reduction before clustering the data. Different approaches for dimensionality reduction have been largely used, such as Principal Components Analysis (PCA) [12], Fastmap [7], Singular Value Decomposition (SVD) [17], and Fractal-based techniques [13, 15]. We have also developed a novel technique named *Nearest-Neighbor Projection* (NNP) for multi-dimensional data exploration on bidimensional spaces [18]. For all these approaches, there is no warranty of dimensionality reduction without losing a considerable amount of information and they are likely to find different clusters in different projections of the same data as shown in the literature [1, 2, 3]. Thus, clustering in projected subspaces could lead to a poor result of the clustering process. Additionally, clustering quality evaluation is dependent on the application. These are the reasons why generating a good clustering, for an specific application, cannot be achieved without direct user interference. Visual clustering techniques exploit this fact by replacing usually costly heuristics with user decisions.

Besides these facts, it is also very desirable for clustering approaches to provide mechanisms for handling outliers<sup>1</sup> and to define a metric for determining whether a cluster is consistent with the user responses.

Table 1 summarizes the features of the most representative approaches of the various families of clustering algorithms<sup>2</sup>. Aggarwal’s IPCLUS algorithm [1], accomplishes almost all the requirements mentioned above. However, IPCLUS cannot be applied to very large data sets due to the costly processes used for projecting the data and estimating the density.

In this work we have developed mechanisms for reducing the time spent in those processes. Those mechanisms were introduced in different steps of the algorithm.

Results demonstrate a processing time reduction of 50% to 92%, as well as clustering improvement in some cases and same clustering quality in all the others. This improve-

<sup>1</sup>Instances that cannot be included in any cluster.

<sup>2</sup>See the work by Berkhin for details on most clustering algorithms found in the literature [4].

	Scalability	Multi-dimensionality	Arbitrary Shape	Outliers	User Interaction	Metric
Birch [21]	√					
Chameleon [14]			√			
Cure [10]	√		√	√		
Bubble [9]	√					
Orclus [3]		√		√		
DBScan [6]	√		√	√		
IPCLUS. [1]		√	√	√	√	√
HD-Eye [11]	√		√	√	√	

**Table 1. Features of representative clustering algorithms.**

ment was measured using the information available about the cluster to which each instance belongs. Software, code, and documentation are open and available.

In the next section we detail the IPCLUS algorithm. In Section 3 we discuss HC-Enhanced, with highlight on the improvements carried out. In Section 4 we present the results on processing time and quality of the clustering generated. Finally, Section 5 is devoted to present our conclusions and future work.

## 2 The IPCLUS algorithm

IPCLUS is a visual density-based algorithm that employs multiple projections to cluster the data admitting user interaction for handling outliers and determining when the process must stop (i.e. when all significant clusters were discovered).

The algorithm can be summarized as follows:

1. Sample  $l$  random data points  $a_i$  and obtain their  $r$ -neighborhood  $\mathcal{M}_i$ .
2. Normalize the groups  $\mathcal{M}_i$  to obtain the groups  $\mathcal{O}_i$ .
3. Determine a bidimensional subspace  $\mathbb{S}$  where the groups  $\mathcal{M}_i$  form well defined clusters.
4. Project the data in subspace  $\mathbb{S}$ .
5. Estimate density in subspace  $\mathbb{S}$ .
6. Specify a noise level  $\lambda$  where the outliers are not included in clusters creation.
7. Create and store the clusters for current projection.
8. Repeat steps 1 through 7 until all clusters were identified.
9. Create final clusters.

User interaction is necessary during steps 6 and 8. In step 6 the user must determine the value of  $\lambda$  through techniques like cutting planes. In step 8 the user decides from the visualization of the results of previous iterations, when all the groups were probably identified and the algorithm must stop. This also will hold for the improved version of the algorithm proposed in this paper.

Each of the  $l$   $\mathcal{M}_i$  groups, where  $l$  is the number of anchors  $a_i$  defined by the user, is formed by the instances in the  $r$ -neighborhood of  $a_i$  (i.e., the result of a range query with center  $a_i$  and radius  $r$ , where  $r$  is user-defined). These groups are normalized such that their centroids<sup>3</sup>  $z_i$  become  $\bar{0}$ . This is achieved subtracting  $z_i$  from all instances in  $\mathcal{M}_i$  generating a normalized group  $\mathcal{O}_i$ .

The bidimensional subspace  $\mathbb{S}$  where  $\mathcal{M}_i$  form well defined groups is that where the moment of all  $\mathcal{O}_i$  is minimal. That is, the subspace where  $\sum_{i=1}^l \mu(\mathcal{O}_i)$  is minimal, where  $\mu(\mathcal{O}_i)$  is the moment of  $\mathcal{O}_i$  defined by the expression  $\mu(\mathcal{O}_i) = \sum_{j=1}^k d(z'_i, m_j)$ .  $d(z'_i, m_j)$  is the distance between the centroid  $z'_i$  of  $\mathcal{O}_i$  and the instance  $(m_j)$ . This is equal to minimizing  $\mu(\cup_{i=1}^l \mathcal{O}_i)$ .

It was proven by Joliffe [12] that the  $d$ -dimensional subspace where a group of data points  $\mathcal{G}$  has the minimum moment is given by the  $d$  eigenvectors associated to the  $d$  smaller eigenvalues of the covariance matrix of  $\mathcal{G}$ . That is, the  $d$  smaller components resulting from applying PCA (Principal Components Analysis) [12] to the set  $\cup_{i=1}^l \mathcal{O}_i$ <sup>4</sup>.

IPCLUS uses this result to find the subspace  $\mathbb{S}$ , incrementally decreasing by a half the dimensionality of the data set.

Once the subspace  $\mathbb{S}$  has been determined, the data set is projected onto it, and the density  $\hat{f}_D$  is estimated using a Gaussian kernel estimator:

$$\hat{f}_D = \frac{1}{nh^d} \sum_i \frac{1}{\sqrt{2\pi}h} e^{-\frac{(x-x_i)^2}{2h^2}} \quad (1)$$

where  $x_i$  is the  $i^{th}$  data point,  $h$  is the smoothing factor,  $n$  the number of instances in the data set, and  $x$  represents a point in the bidimensional Cartesian grid where the density is stored.

The density estimation is mapped into a heightmap presented to the user. The user then specifies a value for the noise level  $\lambda$  such that the outliers will not be included in the clusters creation for the current iteration (projection). As mentioned before, this is done using interaction techniques such as cutting planes. Each cluster is assigned with

<sup>3</sup>The centroid of a group could be thought as being its center of mass.

<sup>4</sup>In our case  $\mathcal{G} = \cup_{i=1}^l \mathcal{O}_i$ .

an identifier that will be added to the *IDString* of each instance belonging to that cluster. The *IDString* of an instance stores the identifiers of the clusters to which the instance belongs in the different iterations (projections).

Once the user decides (based on the visualization) that all the clusters could have been identified, the final clusters are created searching for proper substrings. Details on the handling of IDStrings can be found in the work that described IPCLUS ([1]).

The next section presents the changes performed to this algorithm, in order to seek improvement of processing time and of clustering quality.

### 3 The HC-Enhanced algorithm

The improvements performed on the IPCLUS algorithm lead to the HC-Enhanced. The improvements were directed to three main processes: the range queries performed for forming the groups  $\mathcal{M}_i$  (step 1), the searching for subspace  $\mathbb{S}$  (step 3), and the density estimation (step 5) which is the key point of the improvement in scalability obtained. Within step 3 we also introduce a mechanism for improving projections in cases where the groups cannot be identified clearly.

#### 3.1 Step 1 - Range queries with Omni-sequential

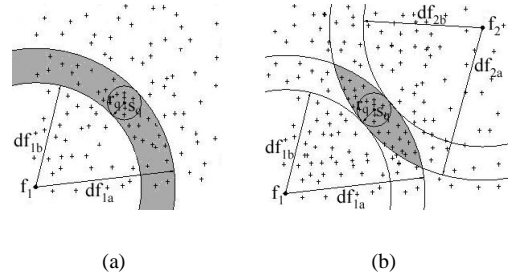
The Omni-sequential is part of a family of data structures proposed by Santos *et al.* [8] that exploits the fact that the intrinsic dimensionality<sup>5</sup> of real data sets is usually considerably low.

The Omni-concept is the base of such family, which basically is a coordinate transformation to the so called Omni-foci base. This base is a set  $\mathcal{F} = \{f_1, f_2, \dots, f_k\}$  of instances belonging to the metric data set to be indexed called foci or focal points. Using this base, the Omni-coordinates are calculated for each instance. The Omni-coordinates of an instance  $s$  are the distances between  $s$  and each  $f_i$ .

The result of using the Omni-concept is twofold. First, the indexing benefits from this “dimensionality reduction” allowing the data structure to index high dimensional data sets without suffering from the “dimensionality curse”. Second, when a range query is performed, a considerable amount of distance calculations is pruned as shown in Figure 1.

It is possible to use this concept together with any data structure. However, when few queries are performed (as in our case), omni-sequential suffices.

<sup>5</sup>The dimensionality of the space actually occupied by the data (a line in  $\mathbb{R}^n$  has intrinsic dimensionality equal to 1).



**Figure 1. A range query with center  $s_q$  and radius  $r_q$ . The shadow illustrates the instances returned from the pruning. (a) Using one focus  $f_1$ . (b) Using two foci  $f_1$  and  $f_2$ .**

#### 3.2 Step 3 - Finding and improving the polarization subspace

In the step 3 of the IPCLUS algorithm, a subspace  $\mathbb{S}$  (the *polarization subspace*) must be found through incrementally decreasing dimensionality by half and projecting the data set. That was done here using PCA.

In some polarization subspaces the groups cannot be identified clearly. Thus, we proposed an approach for improving such projections [18]. In Figure 2 we show that this will generate a better projection when applied to the results of techniques such as Fastmap [7] or PCA [12]. The technique is called the Force scheme and is based on approaching points that should have been projected closer and repelling points that must be further apart from each other (see Figure 2).

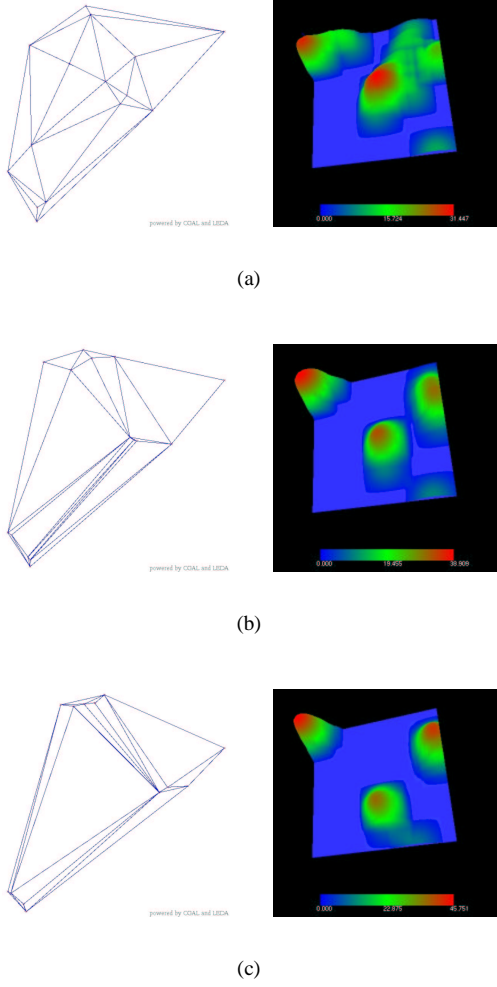
This approach can be used interactively by the user over the projection generated in each iteration (within step 3 of the algorithm), generating a new projection and, thus, a new density map. The improvement in density map generation is described in next subsection.

#### 3.3 Step 5 - Density estimation

This step is the critical point of the algorithm. Every previous steps of the algorithm has complexity  $O(kn)$ , where  $k$  is a constant and  $n$  the number of instances. However, contrary to all other steps, in the density estimation  $k$  is a high constant (the number of grid points that will hold the density).

We propose an approach for reducing the constant  $k$  based on the individual contribution of each instance to the density accumulated in the grid points.

In Figure 3 we present the contribution  $\hat{f}_D^i$  of an instance  $x_i$  in the calculation of the density estimation in a grid point  $x$  as a function of the distance between  $x$  and  $x_i$  for different



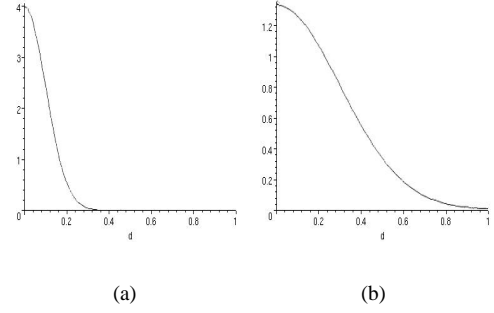
**Figure 2. Improvement of a projection of a synthetic dataset of 15 5-dimensional instances. (a) Iteration 1. (b) Iteration 13. (c) Iteration 25.**

values of the smoothing factor  $h$ . Such contribution is given by the expression:

$$\hat{f}_D^i = \frac{1}{\sqrt{2\pi}h} e^{-\frac{(x-x_i)^2}{2h^2}} \quad (2)$$

We can see that, after some distance, and depending on the smoothing factor, the contribution of instance  $x_i$  to the calculation of the accumulated density in the grid point could be neglected.

Due to the fact that the data set is normalized to the range  $[0, 1]$  in all dimensions during the projection, the smoothing factor  $h$  is between 0 and 1, being that for the data sets tested the optimal  $h$  factor from the formula presented by Joliffe [12] is about 0.1. We can see that, for the case of the optimal



**Figure 3. Contribution of an instance  $x_i$  in the calculus of the density accumulated in a grid point  $x$  as function of the distance  $d$  between  $x$  and  $x_i$  for (a)  $h = 0, 1$  and (b)  $h = 0, 3$ .**

value of  $h$  (0.1), the contribution of an instance to the calculus of the density accumulated in a grid point decreases rapidly as the distance increases.

Thus, we define a neighborhood  $v$  of the grid cells for which an instance will be considered in the calculus of the density accumulated in them.

The approach proposed for calculating the density estimation is as follows.

---

For each instance  $x_i$  with bidimensional coordinates  $x_{ix}$  and  $x_{iy}$ .

1.  $index_x = \lfloor x_{ix} \cdot m \rfloor$ .
  2.  $index_y = \lfloor x_{iy} \cdot m \rfloor$ .
  3. Check for boundary conditions  
 $(index_x + v + 1 > m,$   
 $index_x - v < 0, index_y + v + 1 > m,$   
or  $index_y - v < 0)$ .
  4. Accumulate the contribution of  $x_i$  for all grid points with index in the range  
 $[index_x - v, index_x + v + 1;$   
 $index_y - v, index_y + v + 1]$
- 

Here  $m$  indicates the number of samples of the grid (i.e., for a grid  $30 \times 30$ ,  $m = 30$ ) and  $v$  is the neighborhood.

The original approach, presented here for comparison, is as follows.

---

For each grid point  $x$

For each instance  $x_i$

Accumulate the contribution of  $x_i$  for  $x$ .

---

The optimal neighborhood depends on the number of samples  $m$  of the grid, as well as in the precision  $p$  re-

quired and the smoothing factor  $h$ . The precision indicates the minimum contribution that an instance must provide to the accumulated density in a grid point for it to be considered relevant for such grid point.

Such relation is given by the expression:

$$v = m\sqrt{-2h^2 \ln(p\sqrt{2\pi}h)} \quad (3)$$

directly derived from equation 2.

Figure 4 shows the results of density estimation for different neighborhoods  $v$  for the Iris data set over a grid  $30 \times 30$ . We can see that with a neighborhood of  $v = 5$  (the optimal neighborhood is  $v = 5$  for a precision of 0.008) the density estimation approximates the density estimated using all grid points as the neighborhood (original approach).

In the next section we present the results of applying both the IPCLUS and the HC-Enhanced algorithms to different data sets.

## 4 Results

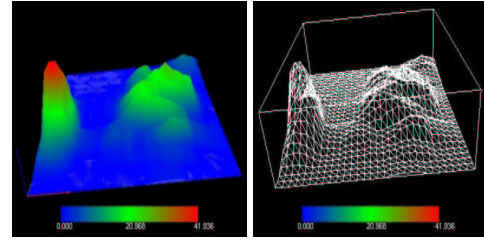
In Figure 5 we present a comparison between the results obtained with the original IPCLUS algorithm and those obtained with the modified version HC-Enhanced in terms of processing time for different sizes of the data set. The tests were performed on a PC computer with a AMD Athlon 1.0 GHz processor and 1.0 GB of memory.

We can see a considerable gain in processing time. These results were obtained without loss in the quality of the clustering. Thus HC-Enhanced is a more interactive and scalable version of the IPCLUS algorithm, which was the main goal of this work.

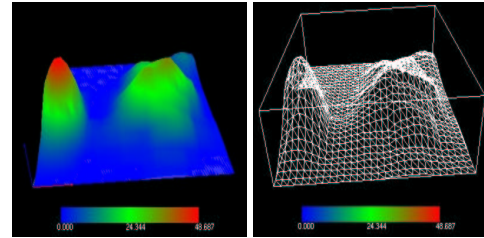
The algorithms were also applied to the Iris and IDH data sets, as well as two synthetic data sets of 1000 10-dimensional and 10000 50-dimensional instances respectively, and to a quadrapeds data set which are described below <sup>6</sup>.

The Iris data set is formed by 150 4-dimensional instances of three classes of plants classified according to the sepal length and width, and the petal length and width. This data set is usually separated by the clustering algorithms in two clusters. One of them formed by the instances of the first class of plants and the other by the two remaining classes. Using the IPCLUS and HC-Enhanced algorithms the first cluster is formed by the 92% of the instances of the first class, whilst the second cluster is formed by the 85% of the instances of the second and third classes. It is important to notice that there were no false positives in the clusters. This result is usually obtained with most clustering algorithms. Besides these two clusters, the HC algorithms

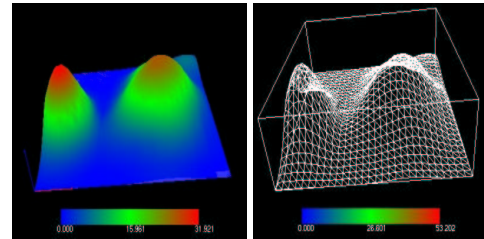
<sup>6</sup>All data sets, with the exception of Synt10d and Synt50d, were taken from the Machine Learning Repository of the University of California, Irvine (<http://www.ics.uci.edu/~mllearn/MLRepository>).



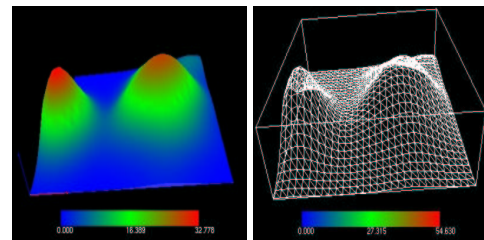
(a)



(b)



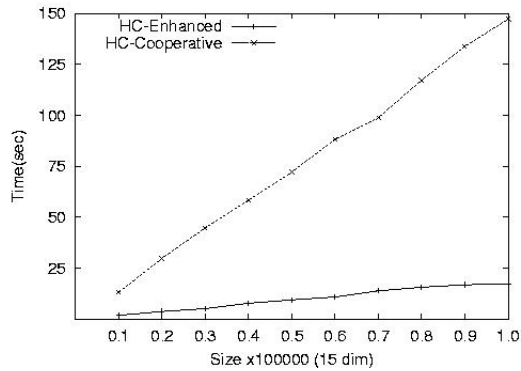
(c)



(d)

**Figure 4. Density estimation for the Iris data set over a grid  $30 \times 30$  with neighborhood (a)  $v = 2$ , (b)  $v = 3$ , (c)  $v = 5$ , and (d)  $v = 30$ .**

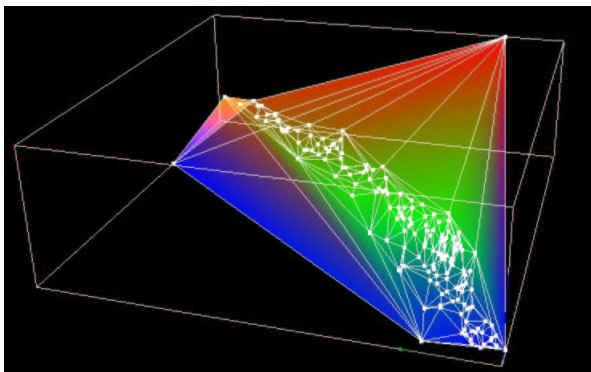
return a cluster formed by 86% of the instances of the second class of plants. Thus, it is possible to infer that there exist three classes of plants in the Iris data set and that two



**Figure 5. Performance comparison between IPCLUS and HC-Enhanced with data sets of different sizes and 15 dimensions.**

of them are very similar. This result cannot be obtained with another clustering algorithm. The HC algorithms are capable of presenting such result due to the nature of the creation of the final clusters process based on multiple projections.

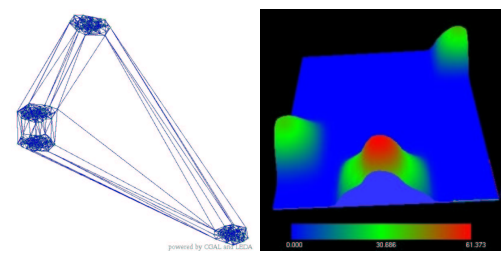
The IDH data set stores data of 173 countries relative to their quality of life. This data set has 9 dimensions plus the ranking of each country, which was not included during the tests. The IPCLUS and HC-Enhanced algorithms separated this data set in two clusters. The first one was formed of the first 25 instances in the ranking. The second cluster was formed by the instances 48 to 173. In Figure 6 we can notice that this result is in accordance with the distribution of the data. There was one false positive in the first cluster and 4 and 9 false negatives in the first and second clusters respectively.



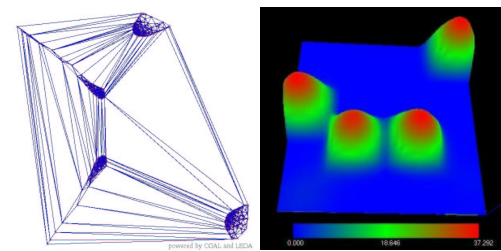
**Figure 6. Delaunay triangulation of the Fastmap projection of the IDH data set. The height and color map the ranking of the country, being that the best ranked is mapped as blue-low and the worst ranked as red-high.**

The first synthetic data set, Synt10d, was created with 1000 10-dimensional instances forming 4 groups, two of them considerably close to each other, and 20% of noise. Applying IPCLUS and HC-Enhanced to this data set led to the following results: four clusters were obtained, the first of them with 88% of the original instances of the first synthetic group, the second with 100% of the instances of the second synthetic group, the third with 88% of the instances of the third synthetic group, and the fourth with 100% of the instances of the last synthetic group, none presenting false positives. The second and third groups presented false negative because they were too close and the instances in the boundaries were assumed to be noise. Normally, only one cluster would be generated from these two groups.

This result was improved using the Force scheme presented in subsection 3.2. The resulting clustering presented 98% (96%, 100%, 96%, 100%, for groups 1 to 4 respectively) of accuracy on average after improving the projection using such scheme. We can see in Figure 7a the four groups with two of them too close to each other. In Figure 7b we show the improved projection, where those groups were separated.



(a)



(b)

**Figure 7. Delaunay triangulation and density estimation calculated over the (a) original and (b) improved projections of a synthetic data set of 1000 10-dimensional instances used for testing the algorithms.**

A second synthetic data set, Synt50d, of 10000 instances and 50 dimensions, was created with three groups and tested with both algorithms. It resulted in a precision of 99.9985%, 99.9973% and 100% for the three clusters respectively.

Finally, a data set formed by 96487 instances grouped in four classes of quadrupeds (dogs, cats, giraffes, and horses) was also used to test both algorithms. Both algorithms generated four clusters. The first of them was formed by 90.80% of the instances of the first class (dogs). From the 25521 instances included in this cluster, 3480 were false positives corresponding to the second class (cats). The second cluster was formed by 70.24% of the instances from the second class (cats). The third cluster was formed by 67.92% of the instances corresponding to the third class (horses). Finally, the fourth cluster was formed by 33.86% of the instances from the third class (giraffes). Neither of the latter three clusters presented false positives. 30.875% of the instances of the original data set were considered noise. This data set is typically difficult to cluster.

Tables 2 and 3 summarize these results, presenting a comparison between the algorithms in relation both to the processing time and the quality of the clustering.

Data set	IPCLUS			
	Average precision	Time	False positives	False negatives
Iris	87,667	0,97s	0	19
IDH	91,907	1,10s	1	13
Synt10d	94,0	6,42s	0	60
Synt50d	99,9986	72,59s	4	10
Quadrupeds	65,518	391,21s	3480	29670

**Table 2. Results of the IPCLUS algorithm on different data sets in terms of precision and processing time.**

Data set	HC-Enhanced			
	Average precision	Time	False positives	False negatives
Iris	87,667	0,14s	0	19
IDH	91,907	0,14s	1	13
Synt10d	98,0	0,93s	0	20
Synt50d	99,9986	20,26s	4	10
Quadrupeds	65,518	201,5s	3480	29670

**Table 3. Results of the HC-Enhanced algorithm on different data sets in terms of precision and processing time.**

We can notice that the only data set to which it was necessary to apply the projection improvement process was Synt10d. The decision of using such improvement was because of the visual results presented to the user during the process, in which it was possible to see that there were clusters too near to each other that should be separated. The rest

of the data sets presented well defined clusters in the original projections or, in the case of the Quadrupeds data set, the metric used was not suitable for separating the two last clusters due to the fact that they were “overlapped” in the Euclidean space. It also affected the performance of the algorithms in terms of processing time, considering that it was the only case where the improvement in the performance was below 70%.

## 5 Conclusion

The improvements to the IPCLUS algorithm reported in this paper led to HC-Enhanced, an algorithm with a better degree of scalability. The gain in processing time obtained from these improvements is considerably high. This reduction exceeded 87% in most of cases. There were only two data sets that presented improvements of 50% and 70%. The quality of the clustering generated by the original algorithm was maintained and the interactivenss of the approach was improved.

The inclusion of faster mechanisms for projecting the data is necessary for obtaining a larger reduction in the processing time.

It is also important to mention that one of the main difficulties found in the density-based clustering is the treatment of categorical data and non-continuous data. The extension of the HC-Enhanced algorithm for clustering categorical data is a work that should be approached in the future.

## 6 Acknowledgements

The authors want to acknowledge the finance of CNPq and FAPESP Brazilian financial agencies.

## References

- [1] C. Aggarwal. A human-computer cooperative system for effective high dimensional clustering. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 221–226, CA, USA, 2001.
- [2] C. Aggarwal. Towards meaningful high-dimensional nearest neighbor search by human-computer interaction. In *ICDE’02, International Conference on Data Engineering*, pages 593–604, San Jose, California, Fevereiro 2002. IEEE Press.
- [3] C. Aggarwal and P. Yu. Finding generalized projected clusters in high dimensional spaces. In W. Chen, J. F. Naughton, and P. A. Bernstein, editors, *ACM SIGMOD International Conference on Management of Data*, pages 70–81, Dallas, Texas, USA, 2000. ACM.

- [4] P. Berkhin. Survey of clustering data mining techniques. Technical report, Accrue Software, San Jose, CA, 2002.
- [5] K. Beyer, J. Goldstein, R. Ramakrishnan, and U. Shaft. When is nearest neighbor meaningful? *Lecture Notes in Computer Science*, 1540:217–235, 1999.
- [6] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In E. Simoudis, J. Han, and U. Fayyad, editors, *Second International Conference on Knowledge Discovery and Data Mining*, pages 226–331, Portland, Oregon, 1996. AAAI Press.
- [7] C. Faloutsos and K. Lin. FastMap: A fast algorithm for indexing, data-mining and visualization of traditional and multimedia databases. In M. J. Carey and D. A. Schneider, editors, *ACM SIGMOD’95 International Conference on Management of Data*, pages 163–174, San Jose, California, 1995. ACM.
- [8] R. F. S. Filho, A. Traina, C. T. Jr, and C. Faloutsos. Similarity search without tears: The omni-family of all-purpose access methods. In *17th IEEE International Conference on Data Engineering (ICDE)*, pages 623–630, Heidelberg, Germany, 2001.
- [9] V. Ganti, M. Ramakrishnan, J. Gehrke, A. Powell, and J. French. Clustering large datasets in arbitrary metric spaces. Technical report, Department of Computer Science, University of Wisconsin at Madison, Madison, Wisconsin, USA, 1998.
- [10] S. Guha, R. Rastogi, and K. Shim. CURE: An efficient clustering algorithm for large databases. In L. M. Haas and A. Tiwary, editors, *ACM SIGMOD International Conference on Management of Data (SIGMOD’98)*, pages 73–84, Seattle, Washington, USA, 1998. ACM Press.
- [11] A. Hinneburg, D. Keim, and M. Wawryniuk. HD-eye: Visual mining of high dimensional data. *IEEE Computer Graphics and Applications*, 19(5):22–31, Setembro/Outubro 1999.
- [12] I. T. Jolliffe. *Principal Component Analysis*. Springer-Verlag, 1986.
- [13] C. T. Jr., A. Traina, L. Wu, and C. Faloutsos. Fast feature selection using fractal dimension. In *Simpósio Brasileiro de Banco de Dados - SBBD’00*, João Pessoa-PA, Brazil, 2000.
- [14] G. Karypis, E. Han, and V. Kumar. CHAMELEON: A hierarchical clustering algorithm using dynamic modeling. *COMPUTER*, 32:68–75, 1999.
- [15] B.-U. Pagel, F. Korn, and C. Faloutsos. Deflating the dimensionality curse using multiple fractal dimensions. In *International Conference on Data Engineering - ICDE’00*, pages 589–598, San Diego-CA, USA, 2000.
- [16] W. Ribarsky, J. Katz, F. Jiang, and A. Holland. Discovery visualization using fast clustering. *IEEE Computer Graphics and Applications*, 19(5):32–39, Setembro/Outubro 1999.
- [17] G. Strang. *Linear Algebra and its Applications*. Academic Press, 1980.
- [18] E. Tejada, R. Minghim, and L. G. Nonato. On improved projection techniques to support visual exploration of multi-dimensional data sets. *Submitted to Information Visualization Journal, Special Issue on Coordinated & Multiple Views in Exploratory Visualization*, 2(4), 2003.
- [19] I. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann, 1999.
- [20] P. C. Wong. Visual data mining. *IEEE Computer Graphics and Applications*, 19(5):20–21, Setembro/Outubro 1999.
- [21] T. Zhang, R. Ramakrishnan, and M. Livny. BIRCH: an efficient data clustering method for very large databases. In *ACM SIGMOD’96 International Conference on Management of Data*, pages 103–114, New York, USA, 1996. ACM.