*First publ. in: First publ. in: Proceedings / Third IEEE International Conference on Data Mining, ICDM 2003 : 19 - 22 November 2003, Melbourne, Florida, pp. 473-476*

# Analyzing High-Dimensional Data by Subspace Validity*

Amihood Amir[†] Reuven Kashi, Nathan S. Netanyahu[‡]
Bar-Ilan University
Department of Computer Science
52900 Ramat-Gan, Israel
{amir,kashi,nathan}@cs.biu.ac.il

Daniel Keim, Markus Wawryniuk
University of Konstanz
Computer & Information Scienc
78457 Konstanz, Germany
{keim,wawryniu}@informatik.uni-konstanz.de

## Abstract

*We are proposing a novel method that makes it possible to analyze high dimensional data with arbitrary shaped projected clusters and high noise levels. At the core of our method lies the idea of subspace validity. We map the data in a way that allows us to test the quality of subspaces using statistical tests. Experimental results, both on synthetic and real data sets, demonstrate the potential of our method.*

## 1. Introduction

The concept of "cluster" is somewhat elusive. From an intuitive sense, it means points in a cluster are "close" to each other while they are "far" from other points. The meaning of "closeness" corresponds to the meaning of "similarity". This definition raises some questions:

**Projected Clusters**: Typically, a relation may exist between some, but not all, variables. Consequently, the clusters are not defined over all attributes, i.e. values in some attributes are similar, but in other attributes not. It is possible that projecting the space into a smaller dimensional space will yield interesting clusters that do not exist in the original data space.

**Topology**: Clusters may have different shapes and linear dependencies. We may be interested in identifying a plane in a multidimensional space as a cluster.

**Dimensions**: A Cluster spreads over a subset of dimensions, implying a relationship between these dimensions. Different clusters may also share some dimensions, meaning that different values of a variable $v$ relate to some subset of variables, while other values of $v$ relate to a different set of variables.

**Overlaps**: It is possible that under a certain projection, two clusters overlap and can not be distinguished, whereas in another projection they are separated. In an even more drastic situation, a cluster may not be identified under *any* projection!

For clustering visual methods have proven to be quite successful. Such methods use the perceptual capabilities of the human. Knowledge about the domain/task flows into the process step by step and is exploited both for a successful understanding of dependencies in the particular domain and for ferring out and separating clusters [6, 7].

Albeit the success of such visualization methods, it is necessary to develop automated clustering algorithms. The reason for this need is the existence of very large high dimensional data sets that need to be analyzed. Considering all different 2- or 3-dimensional projections is clearly not a feasible number for interactive human analysis.

Automated methods have the advantage of speed but the disadvantage of lacking domain knowledge. Harnessing such knowledge through preprocessing work or by machine learning methods is a laborious process and the state-of-the-art is far from satisfactory. The above mentioned challenges of cluster finding do not have good solutions by current methods of automatic data exploration.

[*] This work was partially funded by the Information Society Technologies programme of the European Commission, Future and Emerging Technologies under the IST-2001-33058 PANDA project (2001-2004).

[†] Partially supported by NSF grant CCR-01-04494, and ISF grant 282/01.

[‡] And: Univiversity of Maryland, Center for Automation Research, College Park, MD 20742

Current methods for cluster finding differ in their requirements of domain knowledge and they require parameters (such as the requested number of clusters) as input for the algorithm. In addition, they depend on the amount of noise in the data and that affects the quality of the results.

## 2. Our Contribution

We are proposing a novel method that makes it possible to analyze high dimensional data with high noise levels. Our method requires no domain knowledge in advance, yet it discovers projected clusters and allows separating overlapping clusters with different topologies.

At the core of our method lies the idea of *subspace validity*. We map the data in a way that allows us to test the parameters of a one-dimensional subspace. It is possible to perform various statistical tests efficiently in one dimension. In these projections, one of the variables is designated as the subspace whose validity is checked by various statistical means. The result of these tests allows us to reach a conclusion about clustering in the low-dimensional subspace.

Our goal is analyzing high-dimensional data sets in order to find structures and patterns that can be considered as interesting to the end user. It is accepted that if the human eye would perceptually capture a pattern in a subset of data points, then it is considered as valuable information which should be noticed and investigated further. The traditional way to capture such “similar” data objects is by the various definitions of clustering in the literature. Therefore, to demonstrate the efficiency and effectiveness of our proposed method, we compare it against clustering algorithms in the databases literature. However, it should be stressed that we are not defining “clusters” in any traditional formal sense. We are seeking a more general method that can automatically detect “interesting” structures in high dimensional data sets. Therefore, our use of the word “clusters” to define such structures is intentionally quite loose.

## 3. The Subspace Validity Algorithm

Given a $m$-dimensional database DB, we first project the data set onto every subset of 3 dimensions. For each 3-dimensional projection, we designate one dimension (i.e., attribute) as the *vertical dimension*, i.e., the one-dimensional subspace whose subspace validity is tested. The proposed approach has four major stages.

**Constructing compact images from the data.** The first stage consists of building compact 2-dimensional images for every triple of dimensions $\langle X, Y, Z\rangle$. Let $M_{XY}$ denote the image matrix. In every $M_{XY}$ entry or pixel $(x, y)$ we store information about the conditional distribution $P(Z|(x, y))$ of the $Z$ variable associated with that entry.

**Feature extraction from the image.** Recall that for the same $(x, y)$ location in the image $M_{XY}$, there can be many $z$ values in the $Z$ dimension. A feature function returns for each pixel an extracted feature vector associated with that pixel. These values are assigned to $M^Z_{XY}(x, y)$. In our implementation we used the Haar wavelet transformation for our feature function.

Let us illustrate this idea with a simple case. In [4], a single representative value was chosen for each entry of $M_{XY}$, the *median* of all $Z$ values at location $(x, y)$, assuming a unimodal distribution at that location.

**Image segmentation.** Since $M^Z_{XY}$ represents an image, we can segment the image by applying standard *image segmentation* techniques. The segmentation of $M^Z_{XY}$ yields regions, such that the pixels of a region have similar features. The strategy we have used for segmentation is based on region growing.

**Region analysis.** At this stage we have possibly a large number of different regions of the various $X$ and $Y$ attributes, such that the values of the associated vertical $Z$ dimensions in each such region are distributed in a similar manner. Using this information about the regions, we can analyze the data points in order to further discover prospective clusters either in the same attribute subspace or in some augmented subspace projections.

The clustering scheme we have used consists of the following characteristics:

**1.** Consider the two-dimensional projection on $X$ and $Y$. A pixel stores information about the distribution of the $Z$ values of the points which are mapped to the corresponding cell.

**2.** We find regions (dense and compact) with similar distribution of $Z$.

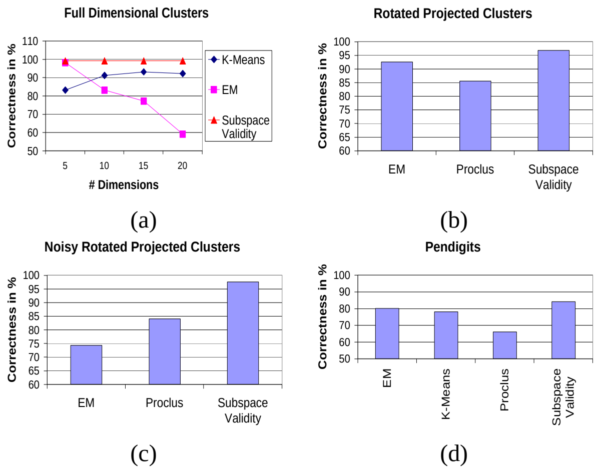
**3.** A region corresponds to a low-dimensional cluster. In order to augment the set of dimensions which define the cluster, we partition all points into two sets; points which belong to the region and points which do not. The two subsets are processed recursively. We use the *partition tree* concept which is a generalization of the separator tree concept used in HD-Eye [6].

**4.** Searching for projected clusters is motivated by the fact that it might not be possible to augment a certain dimension set, i.e., the cluster is not defined in all dimensions; it is a projected cluster.

**5.** Clusters correspond to the leafs of the partition tree.

## 4. Evaluation and Comparison

In the experiments, we use a number of data sets with controlled characteristics, such as the number of dimensions or noise level, as well as real data sets. In the exper-



(a) (b)

(c) (d)

**Figure 1. Experimental results**

iments, we compare our new approach to existing methods including Expectation Maximization (EM) [5], K-Means and PROCLUS [1].

In the first experiment, we present the effectiveness of our approach on clusters which are defined in all dimensions, which we will refer to as full-dimensional clusters. The type of data we generated is similar to the data created in [1], taking into account that all dimensions are selected. The results are shown in Figure 1(a). The performance of K-Means remains constant and EM is degenerating with increasing dimensionality, whereas our solution provides the best overall effectiveness.

Full-dimensional clusters are unlikely to occur in high-dimensional data sets. Real clusters often describe linear dependencies between the dimensions which define the cluster. Therefore we analyze the ability of the algorithms to find rotated projected clusters. The type of data we generated is similar to the data created in [1] and [2]. The results are shown in Figure 1(b). The results represent the average correctness over a number of data sets of this type. Subspace Validity has the best overall correctness. EM performs surprisingly well and provides a better performance than PROCLUS.

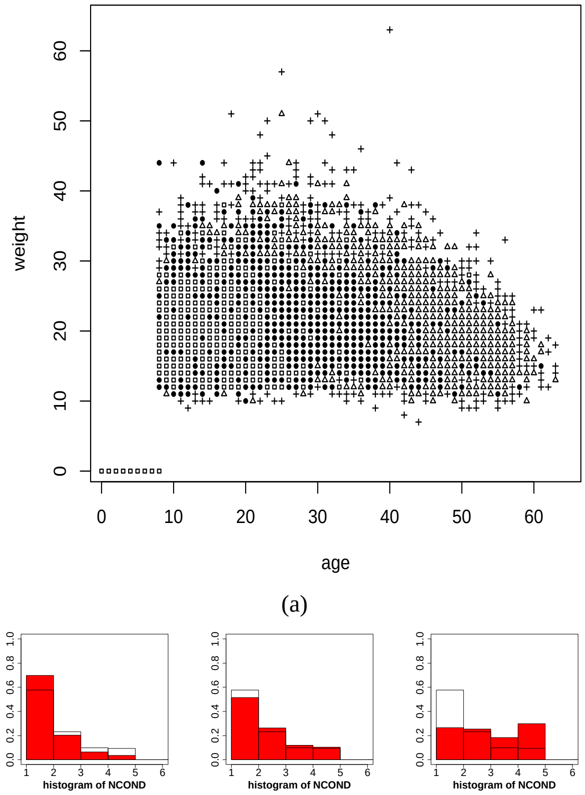
In the next experiment, we examined the effect of noise in the data. For this series of experiments, we added random noise to the data sets from our second experiments and repeated the tests. The new results (see Figure 1(c)) show that the effectiveness of our method remains unchanged, but the performance of EM and PROCLUS degenerates. Interestingly, the performance of EM degenerates much faster than the performance of PROCLUS.

We performed experiments with two real data sets. The first one is the *pendigits* data set from the University of California at Irvine's Machine Learning Repository (www.ics.uci.edu/∼mlearn/MLRepository.html). The pendigits data set contains 7,494 tuples and 16 dimensions that describe handwritten digits. The results of this experiment is shown in Figure 1(d). With our method we get the best correctness of 84%, followed by EM with 80% and K-Means with 78%. From the extremely low classification rate of PROCLUS (only 66%, we tested all kinds of values for the number of bounded dimensions), we may conclude that clusters are spread over all dimensions.

Our second real data set is the census data set *nhis93ac* (NHIS – National Health Interview Survey 1993). The data set is available from (http://ferret.bls.census.gov/). It has several hundred numeric and nominal attributes and consists of 45951 records. The numerical attributes *AGE*, *BDDAY12*, *DV12*, *EDUC*, *INCFAMR*, *NCOND*, *WEIGHT* were selected. Since the data does not come with a known classification, we have to judge the accuracy of the clusters by looking at the results. For this purpose, we use the Subspace Validity plots.

An example can be seen in Figure 2. The clusters are defined in three dimensions, namely AGE, WEIGHT and NCOND. The AGE attribute is on the horizontal axis, WEIGHT is the vertical axis, and NCOND is the vertical dimension and is represented by a symbol.[1] Recall that a region in the $X, Y$ projection with the same symbol means that the pixels belonging to the region have a similar data distribution in the vertical dimension $Z$. Therefore, same symbol means a cluster with similar data points in the third attribute. The clusters can be easily understood with the histogram of $Z$. Those histograms are shown in Figure 2(b),(c) and (d). The histogram is shown with gray bars, and the black line indicates the histogram of the full data set. One can see that the data points falling in the region marked with squares have lower values compared to the full data set, because the leftmost bin of the histogram for the region in Figure 2(b) is more than 10% higher than the histogram corresponding to the full dataset. Analogously, one can see that points falling in the region with solid circles have a similar distribution in the $Z$ attribute as the full data set. But points falling in the region with the triangle point-up have higher values compared to the full data set. The pixels labeled with the plus sign are outliers. They belong to regions with very low support. The number of points falling in a particular region is 22946, 17509 and 5182, respectively. The clusters found by our method tell a very interesting story. The diagonal in the projected cluster from top left to bottom right indicates something that insurance companies love to find. The number of medical conditions increases at a younger age when the weight is greater. In fact, the diagonal can even predict at what weight and at what age one can expect the problems to accumulate.

[1] In reality we use colors, which will increase considerably the ability to recognize the regions.



(a)

(b) (c) (d)

**Figure 2. An example from a census data set**

## 5. Related Work

Cluster finding has been an extensively studied problem for many years by the statistics, machine learning and database communities. For the *full-dimensional* case many clustering algorithms have been proposed [8].

The idea of *projected clustering* has attracted a lot of attention during the last few years. One of the first algorithms dealing with projected clustering is CLIQUE [3]. The algorithm mines the projection space bottom up by searching frequent combinations of histogram bins which are assembled to clusters on a single linkage basis.

The algorithms PROCLUS [1] and ORCLUS [2] are $k$-means like algorithms. Each cluster found is described by a centroid and a set of vectors, spanning the subspace of the projected cluster. PROCLUS reduces the full-dimensional data space to the subspace spanned by the dimensions with the smallest variance (which are treated independently) with the result that only axes-parallel projected clusters can be found. In contrast, ORCLUS determines for each cluster the eigenvectors of the covariance matrix with the smallest eigenvalues and therefore allows arbitrary orientations.

The most recent method DOC [9] defines a projected cluster as a hyperbox, with a boundary size of $w$ in the bounded dimensions and an unbounded size in the other dimensions. DOC uses sampling to center the boxes, and an optimal projected cluster maximizes a quality function.

## 6. Conclusions and Future Work

We have shown a methodology for finding projected clusters in high dimensional data sets. Our idea resulted in an effective way of automatically considering low dimensional projections of the data set and analyze them appropriately to obtain meaningful projected clusters. Our method constructs higher dimensional clusters from data on small dimensional projections of those clusters. In future research we plan to pursue faster implementation for our approach as well as extending it into a visual data mining system.

## References

[1] C. C. Aggarwal, C. M. Procopiuc, J. L. Wolf, P. S. Yu, and J. S. Park. Fast algorithms for projected clustering. In *Proc. ACM SIGMOD International Conference on Management of Data, June 1-3, 1999, Philadephia, Pennsylvania, USA*, pages 61–72. ACM Press, 1999.

[2] C. C. Aggarwal and P. S. Yu. Finding generalized projected clusters in high dimensional spaces. In *Proc. of the 2000 ACM SIGMOD International Conference on Management of Data, May 16-18, 2000, Dallas, Texas, USA*, pages 70–81. ACM, 2000.

[3] R. Aggrawal, J. Gehrke, D. Gunopulos, and P. Raghavan. Automatic subspace clustering of high dimensional data for data mining applications. In *Proc. ACM SIGMOD International Conference on Management of Data, June 2-4, 1998, Seattle, Washington, USA*, pages 94–105. ACM Press, 1998.

[4] A. Amir, R. Kashi, and N. S. Netanyahu. Analyzing quantitative databases: Image is everything. In *VLDB 2001, Proc. of 27th International Conference on Very Large Data Bases*, pages 89–98, Roma, Italy, September 11-14 2001.

[5] A. P. Dempster, N. Laird, and D. Rubin. Maximum likelihood for incomplete data via the EM algorithm. *J. of the Royal Statistical Society, ser. B*, 39:1–38, 1977.

[6] A. Hinneburg, M. Wawryniuk, and D. A. Keim. Hd-eye: Visual mining of high-dimensional data. *IEEE Computer Graphics & Applications Journal*, 19(5):22–31, September 1999.

[7] D. A. Keim. Information visualization and visual data mining. *IEEE Transactions on Visualization and Computer Graphics (TVCG)*, 8(1):1–8, January–March 2002.

[8] D. A. Keim and A. Hinneburg. Clustering techniques for large data sets - from the past to the future. In *Tutorial Notes for ACM SIGKDD 1999 International Conference on Knowledge Discovery and Data Mining*, pages 141–181, San Diego, CA, 1999. ACM Press.

[9] C. M. Procopiuc, M. Jones, P. K. Agarwal, and T. M. Murali. A monte carlo algorithm for fast projective clustering. In *Proc. of the ACM SIGMOD international conference on Management of data*, pages 418–427. ACM Press, 2002.