Mining Hierarchies of Correlation Clusters

Elke Achtert, Christian Böhm, Peer Kröger, Arthur Zimek
Institute for Computer Science, University of Munich, Germany
{achtert,boehm,kroegerp,zimek}@dbs.ifi.lmu.de

Abstract

The detection of correlations between different features in high dimensional data sets is a very important data mining task. These correlations can be arbitrarily complex: One or more features might be correlated with several other features, and both noise features as well as the actual dependencies may be different for different clusters. Therefore, each cluster contains points that are located on a common hyperplane of arbitrary dimensionality in the data space and thus generates a separate, arbitrarily oriented subspace of the original data space. The few recently proposed algorithms designed to uncover these correlation clusters have several disadvantages. In particular, these methods cannot detect correlation clusters of different dimensionality which are nested into each other. The complete hierarchical structure of correlation clusters of varying dimensionality can only be detected by a hierarchical clustering approach. Therefore, we propose the algorithm HiCO (Hierarchical Correlation Ordering), the first hierarchical approach to correlation clustering. The algorithm determines the cluster hierarchy, and visualizes it using correlation diagrams. Several comparative experiments using synthetic and real data sets show the performance and the effectivity of HiCO.

1. Introduction

The detection of correlations between different features in a given data set is a very important data mining task. High correlation of features may result in a high degree of collinearity or even a perfect one. Thus, strong correlations between different features correspond to approximate linear dependencies between two or more attributes. These dependencies can be arbitrarily complex, one or more features might depend on a combination of several other features. In the data space they appear as lines, planes, or, generally speaking, hyperplanes exhibiting a relative high density of data points compared to the surrounding space. Knowledge about correlations enables the user to detect hidden causalities and to reduce the dimensionality of the data set.

Correlation clustering [6] is a novel concept of knowledge discovery in databases which has been successfully applied to applications like target marketing in recommendation systems, gene expression analysis in molecular biology, metabolome data analysis for individual drug design, etc. It corresponds to the marriage of two widespread ideas: Correlation analysis, typically by a principal component analysis (PCA) and clustering, i.e. identifying local subgroups of data objects sharing similar properties. Correlation clustering groups the data sets into subsets called correlation clusters such that the objects in the same correlation cluster are all associated to the same hyperplane of arbitrary dimensionality. We will refer to a correlation cluster associated to a λ-dimensional hyperplane as a *λ-dimensional correlation cluster*. The dimensionality of a hyperplane associated to a correlation cluster is called *correlation dimensionality*.

In [6] it was demonstrated that a trivial combination of the concepts of clustering and correlation detection is not sufficient to find correlation clusters. Therefore, algorithms for correlation clustering have to integrate the concepts of clustering and correlation detection in a more sophisticated way. The algorithm ORCLUS [2], for instance, integrates PCA into k-means clustering and the algorithm 4C [6] integrates PCA into the density based clustering algorithm DBSCAN [8]. Both algorithms decompose a data set into subsets of points, each subset being associated to a specific λ-dimensional hyperplane. Since both methods use the correlation dimensionality λ as a global parameter, i.e. the correlation dimensionality of the resulting clusters must be determined by the user, they are both not able to find all correlation clusters of different dimensionality during one single run.

Moreover, searching clusters of different dimensionality is essentially a *hierarchical problem* because sev-

Proceedings of the 18th International Conference on Scientific and Statistical Database Management (SSDBM'06)
0-7695-2590-3/06 $20.00 © 2006 IEEE

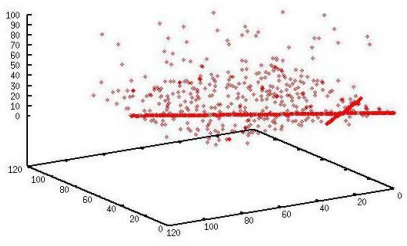


Figure 1. Hierarchies of correlation clusters.

eral correlation clusters of low dimensionality may together form a larger correlation cluster of higher dimensionality, and so on. For example, consider two lines in a 3D space that are embedded into a 2D plane (cf. Figure 1). Each of the two lines forms a 1-dimensional correlation cluster. On the other hand the plane is a 2-dimensional correlation cluster that includes the two 1-dimensional correlation clusters. In order to detect the lines, one has to search for 1-dimensional correlation clusters, whereas in order to detect the plane, one has to search for 2-dimensional correlation clusters.

None of the previously proposed algorithms for correlation clustering is able to detect hierarchies of correlation clusters. Therefore, in this paper we propose HiCO (Hierarchical Correlation Ordering), a new algorithm for searching simultaneously for correlation clusters of arbitrary dimensionality and detecting hierarchies of correlation clusters. Additionally, HiCO overcomes another drawback of the existing non-hierarchical correlation clustering methods like 4C and ORCLUS, since HiCO does not require the user to define critical parameters that limit the quality of clustering such as a density threshold or the number of clusters in advance.

The remainder of this paper is organized as follows: Section 2 introduces the related work. Section 3 explains the necessary preliminaries to be performed before applying our new algorithm HiCO. In section 4 we define the notion of hierarchical correlation clusters and introduce the algorithm HiCO. Section 5 contains an extensive experimental evaluation of HiCO. Section 6 concludes the paper.

2. Related Work

As most traditional clustering algorithms usually fail to detect clusters in high-dimensional data, several subspace clustering algorithms have been proposed recently. However, subspace clustering algorithms [3, 1, 13, 10, 5] only find axis parallel projections of the data. Since the principal axes of correlated data are arbitrarily oriented, subspace clustering algorithms are not able to capture local data correlations and find clusters of correlated objects.

Pattern-based clustering methods [16, 15, 12, 11] aim at grouping objects that exhibit a similar trend in a subset of attributes into clusters rather than objects with low distance. This problem is also known as co-clustering or biclustering [9, 7]. In contrast to correlation clustering, pattern-based clustering is limited to a special form of correlation where all attributes are positively correlated. It does not capture negative correlations nor correlations where one attribute is determined by two or more other attributes. Thus, bi-clustering or pattern-based clustering could be regarded as a special case of correlation clustering [6].

ORCLUS [2] integrates PCA into k-means clustering. In each iteration of the k-means algorithm, the association of points to correlation clusters is optimized, and the correlation clusters are recomputed under the side condition that the overall average dimensionality of all hyperplanes associated to the respective correlation clusters corresponds to a user-provided parameter λ. The resulting correlation clusters are allowed to exist in arbitrarily oriented subspaces represented by a set of Eigenvectors. Beside the parameter λ, ORCLUS requires the user to specify the number of clusters k in advance. ORCLUS usually has problems with correlation clusters of very different dimensionalities because the resulting clusters must have dimensionalities near λ.

4C [6] integrates PCA into DBSCAN [8] and searches for arbitrary linear correlations of fixed dimensionality. The algorithm assigns to each point P a similarity matrix which is determined from the covariances of the vectors in the neighborhood of P. The point P is said to be a core point of a correlation cluster, if the points in its neighborhood form a λ-dimensional hyperplane where λ is a user-defined parameter. The neighboring points may also deviate from the ideal plane up to a certain degree which is also a user defined parameter. Two correlation core points are united into a common correlation cluster if the associated hyperplanes are sufficiently similar. Beside the dimensionality λ of the computed correlation clusters, the user must specify additional parameters to define the minimum density of a cluster. 4C may also miss important clusters for similar reasons as ORCLUS. In addition, the global density threshold for correlation clusters specified by the input parameters is often also hard to determine since correlation clusters of different dimensionality will most likely exhibit different densities.

Quite recently, CURLER, a method to detect arbi-

Proceedings of the 18th International Conference on Scientific and Statistical Database Management (SSDBM'06)
0-7695-2590-3/06 $20.00 © 2006 IEEE

trary, non-linear correlations, has been proposed [14]. CURLER generates micro-clusters using an EM variant which are then merged to uncover correlation clusters. CURLER improves over ORCLUS and 4C as the correlations underlying the clusters are not necessarily linear. In addition, by merging several clusters according to their co-sharing level the algorithm on the one hand becomes less sensitive to the predefined number of clusters k. On the other hand, the user becomes disabled to directly derive a model describing the correlations, since the original k models are no longer persistent in the resulting clustering.

Let us note that none of the proposed approaches to correlation clustering can detect hierarchies of correlation clusters, i.e. lower-dimensional correlations within a common higher-dimensional correlation.

3. Preliminaries

To determine how two points are correlated, we introduce in section 4 a special distance measurement called *correlation distance*. This measurement is based on the *local correlation dimensionality* of a point which reflects the dimensions having a strong correlation in the neighborhood of this point. In order to compute the correlation distance between two points we have to determine in a preprocessing step for each point P of the data set:

1. The *local covariance matrix* $\mathbf{\Sigma}_P$ which is the covariance matrix of the k nearest neighbors of P.
2. The *local correlation dimensionality* λ_P which indicates the highly correlated dimensions in the k nearest neighbors of P.
3. The *local correlation similarity matrix* $\hat{\mathbf{M}}_P$ which is used to compute the local correlation distance between two points.

In the following, $\mathcal{D}$ denotes a set of points and *dist* is the Euclidean distance function.

Definition 1 (local covariance matrix)
Let $k \in \mathbb{N}$, $k \leq |\mathcal{D}|$. The local covariance matrix $\mathbf{\Sigma}_P$ *of a point $P \in \mathcal{D}$ w.r.t. k is formed by the k nearest neighbors of P.*

Formally[1]*: Let $\overline{X}$ be the centroid of $NN_k(P)$, then*

$$\mathbf{\Sigma}_P = \frac{1}{|NN_k(P)|} \cdot \sum_{X \in NN_k(P)} (X - \overline{X}) \cdot (X - \overline{X})^{\mathbf{T}}$$

[1]Note that we use column vectors by convention. Therefore, $(X - \overline{X}) \cdot (X - \overline{X})^{\mathbf{T}}$ is not the inner (scalar) product but the outer product constructing a $d \times d$ matrix.

Since the local covariance matrix $\mathbf{\Sigma}_P$ of a point P is a square matrix it can be decomposed into the Eigenvalue matrix $\mathbf{E}_P$ *of P and the* Eigenvector matrix $\mathbf{V}_P$ *of P such that $\mathbf{\Sigma}_P = \mathbf{V}_P \cdot \mathbf{E}_P \cdot \mathbf{V}_P^{\mathbf{T}}$.*

The Eigenvalue matrix $\mathbf{E}_P$ is a diagonal matrix holding the Eigenvalues of $\mathbf{\Sigma}_P$ in decreasing order in its diagonal elements. The Eigenvector matrix $\mathbf{V}_P$ is an orthonormal matrix with the corresponding Eigenvectors of $\mathbf{\Sigma}_P$.

Unlike in [6] we do not base the local covariance matrix on a range query predicate because for our hierarchical clustering method, we do not have a predefined query radius and there exists no natural choice for such a radius. Therefore, we prefer to base the local correlation on a certain number k of nearest neighbors which is more intuitive.

Now we define the local correlation dimensionality of a point P as the number of dimensions of the (arbitrarily oriented) subspace which is spanned by the major axes of the k nearest neighbors of P. If, for instance, the points in a certain region of the data space are located near by a common line, the correlation dimensionality of these points will be 1. That means we have to determine the principal components of the points in $NN_k(P)$. The Eigenvector associated with the largest Eigenvalue has the same direction as the first principal component, the Eigenvector associated with the second largest Eigenvalue determines the direction of the second principal component and so on. The sum of the Eigenvalues equals the trace of the square matrix $\mathbf{\Sigma}_P$ which is the total variance of the points in $NN_k(P)$. Thus, the obtained Eigenvalues are equal to the variance explained by each of the principal components, in decreasing order of importance. The correlation dimensionality of a point P is now defined as the smallest number of Eigenvectors explaining a portion of at least α of the total variance of the k nearest neighbors of P:

Definition 2 (local correlation dimensionality)
Let $\alpha \in]0, 1[$. Then the local correlation dimensionality λ_P *of a point P is the smallest number r of Eigenvalues e_i in the $d \times d$ Eigenvalue matrix $\mathbf{E}_P$ for which*

$$\frac{\sum_{i=1}^{r} e_i}{\sum_{i=1}^{d} e_i} \geq \alpha$$

We call the first λ_p Eigenvectors of $\mathbf{V}_P$ strong *Eigenvectors, the other Eigenvectors are called* weak.

Typically α is set to values between 0.80 and 0.90, e.g. $\alpha = 0.85$ denotes that the obtained principal components explain 85% of the total variance. In the following we denote the λ_P-dimensional subspace which

Proceedings of the 18th International Conference on Scientific and Statistical Database Management (SSDBM'06)
0-7695-2590-3/06 $20.00 © 2006 IEEE

is spanned by the major axes of the neighborhood of P the *correlation hyperplane* of P.

In the third step of the preprocessing phase we associate each point with a so-called local similarity matrix which is used to compute the local correlation distance to another point of the data set. The local similarity matrix can be derived from the local covariance matrix in the following way:

Definition 3 (local correlation similarity matrix) *Let point $P \in \mathcal{D}$, $\mathbf{V}_P$ the corresponding $d \times d$ Eigenvector matrix of the local covariance matrix $\mathbf{\Sigma}_P$ of P, and λ_P the local correlation dimensionality of P. The matrix $\hat{\mathbf{E}}_P$ with entries $\hat{e}_i$ ($i = 1, \ldots, d$) is computed according to the following rule:*

$$\hat{e}_i = \begin{cases} 0, \textit{ if } i \leq \lambda_P \\ 1, \textit{ otherwise} \end{cases}$$

The matrix $\hat{\mathbf{M}}_P = \mathbf{V}_P \hat{\mathbf{E}}_P \mathbf{V}_P^{\mathbf{T}}$ is called the local correlation similarity matrix *of P.*

Definition 4 (local correlation distance) *The* local correlation distance *of point P to point Q according to the local correlation similarity matrix $\hat{\mathbf{M}}_P$ associated with point P is denoted by*

$$\textsc{LocDist}_P(P, Q) = \sqrt{(P - Q)^{\mathbf{T}} \cdot \hat{\mathbf{M}}_P \cdot (P - Q)}.$$

$\textsc{LocDist}_P(P, Q)$ is the weighted Euclidean distance between P and Q using the local correlation similarity matrix $\hat{\mathbf{M}}_P$ as weight. The motivation for the adaptation of $\hat{\mathbf{M}}_P$ is that the original local covariance matrix $\mathbf{\Sigma}_P$ has two undesirable properties: (1) It corresponds to a similarity measure and to an ellipsoid which is oriented perpendicularly to the major axes in the neighborhood of P. This would result in high distances to points lying within or nearby the subspace of the major axes and in low distances to points lying outside. Obviously, for detecting correlation clusters we need quite the opposite. (2) The Eigenvalues vary with the data distribution, so some points P may have higher Eigenvalues in $\mathbf{E}_P$ than others and this would lead to incomparably weighted distances. Thus, to compute comparable local correlation distances an inversion and a scaling of the Eigenvalues has to be performed. Intuitively spoken, the local correlation distance $\textsc{LocDist}_P(P, Q)$ equals the Euclidean distance between Q and the correlation hyperplane exhibited by the neighbors of P. Thus, if Q lies within the correlation hyperplane of P then $\textsc{LocDist}_P(P, Q) = 0$.

We call the matrix $\hat{\mathbf{E}}$ the *selection matrix of the weak Eigenvectors* because $\mathbf{V}_P \cdot \hat{\mathbf{E}}$ provides a matrix containing only weak Eigenvectors. We will later also use another matrix $\check{\mathbf{E}}_P = \mathbf{I}_{d \times d} - \hat{\mathbf{E}}_P$ where the 0 and 1-entries of the diagonal elements are changed. We call this matrix $\check{\mathbf{E}}$ the *selection matrix of the strong Eigenvectors* since $\mathbf{V}_P \cdot \check{\mathbf{E}}$ provides a matrix containing only strong Eigenvectors. The selection matrices $\hat{\mathbf{E}}_P$ and $\check{\mathbf{E}}_P$ only depend on the local correlation dimensionality λ_P: $\hat{\mathbf{E}}_P$ is a $d \times d$ diagonal matrix where the first λ_P diagonal elements are 0 and the remaining $d - \lambda_P$ diagonal elements are 1 (and *vice versa* for $\check{\mathbf{E}}_P$).

Note that the local correlation distance is not yet the similarity measure which is directly used in our hierarchical clustering algorithm. It is merely a construction element for the actual correlation distance measure which will be defined in the following section.

4. Hierarchical Correlation Clusters

Hierarchical clustering methods in general are able to find hierarchies of clusters which are nested into each other, i.e. weaker clusters in which some stronger clusters are contained. The hierarchical density based clustering method OPTICS, for example, is able to detect clusters of higher density which are nested in clusters of lower but still high density.

The task of correlation clustering as defined in [6] is to group those points of a data set into same clusters where the correlation is uniform. Our general idea is to evaluate the correlation between two points with a special distance measure called *correlation distance*. This distance results in a small value whenever many attributes are highly correlated in the neighborhood of the two points. In contrast, the correlation distance is high if only a few attributes are highly correlated or the attributes are not correlated at all. Therefore, our strategy is to merge those points into common clusters which have small correlation distances. A hierarchy of correlation clusters means that clusters with small correlation distances (e.g. lines) are nested in clusters with higher correlation distances (e.g. 2d-planes).

4.1. Main Concepts of HiCo

Once we have associated the points of our database with a local correlation dimensionality and with a decomposed local similarity matrix, we can now explain the main idea of our hierarchical clustering algorithm. Conventional hierarchical clustering algorithms like SLINK or OPTICS without the idea of correlation work as follows: They keep two separate sets of points, points which have already been placed in the cluster structure and those which have not. In each step, one point of the latter set is selected and placed in the first set. The algorithm always selects that point

Proceedings of the 18th International Conference on Scientific and Statistical Database Management (SSDBM'06)
0-7695-2590-3/06 $20.00 © 2006 IEEE

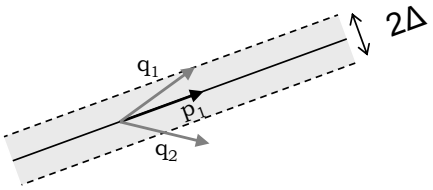


Figure 2. Spaces spanned by two vectors.

which minimizes the distance to any of the points in the first set. By this selection strategy, the algorithm tries to extend the current cluster hierarchy as close to the bottom of the hierarchy as possible.

We will adapt this paradigm. In the context of hierarchical correlation clustering, the hierarchy is a containment hierarchy of the correlation primitives. Two or more correlation lines may together form a 2-dimensional correlation plane and so on. We simulate this behavior by defining a similarity measure between points which assigns a distance of 1, if these two points (together with their associated similarity matrices) share a common correlation line. If they share a common correlation plane, they have a distance of 2, etc. Sharing a common plane can mean different things: Both points can be associated to a 2-dimensional correlation plane and the planes are the same, or both points can be associated to 1-dimensional correlation lines and the lines meet at some point or are parallel (but not skew).

If we associate a pair of points with a distance measure with the properties mentioned before, we can generally use the well-known hierarchical clustering algorithms. Intuitively, the distance measure between two points corresponds to the dimensionality of the data space which is spanned by the strong Eigenvectors of the two points. By the notion *spanning a space* we do not mean spanning in the algebraic sense of linear independence which considers two vectors to span a 2-dimensional space even if they have only a minimal difference of orientation. In our context, a vector q adds a new dimension to the space spanned by a set of vectors $\{p_1, \ldots, p_n\}$ if the "difference" between q and the space spanned by $\{p_1, \ldots, p_n\}$ is substantial, i.e. if it exceeds the threshold parameter Δ. This is illustrated in Figure 2: the space spanned by $\{q_1\} \cup \{p_1\}$ is considered to be the same as the space spanned by p_1 only. On the other hand, the set of vectors $\{q_2\} \cup \{p_1\}$ span a 2-dimensional space, as the "difference" between q_2 and p_1 exceeds Δ.

We first give a definition of the *correlation dimensionality of a pair of points* $\lambda(P, Q)$ which follows the intuition of the spanned space. Later we will give a method for computing this dimensionality efficiently, given the local Eigenvector matrices and the local correlation dimensionalities of P and Q, respectively. The correlation dimensionality is the most important component of our correlation similarity measure which will later be extended a little bit one more time.

Definition 5 (correlation dimensionality) *The* correlation dimensionality *between two points $P, Q \in \mathcal{D}$, denoted by $\lambda(P, Q)$, is the dimensionality of the space which is spanned by the union of the strong Eigenvectors associated to P and the strong Eigenvectors associated to Q.*

If we would like to determine the correlation dimensionality of two points P and Q in a strong algebraic sense, we would have to look *exactly* for the linearly independent vectors in the union of the strong Eigenvectors of P and Q. These n vectors form a basis of the n-dimensional subspace spanned by the strong Eigenvectors of P and Q. Note that we are not interested in the spanned space in the strong algebraic sense. That means we are not looking for vectors that are linearly independent in strict algebraic sense but only for those vectors that are linearly independent in our relaxed notion as mentioned above in order to allow a certain degree of jitter of data points around a perfect hyperplane.

An obvious idea for computing the correlation dimensionality of a pair of objects is to compare the *strong* Eigenvectors $p_i \in \mathbf{V}_P \cdot \check{\mathbf{E}}_P$ and $q_i \in \mathbf{V}_Q \cdot \check{\mathbf{E}}_Q$ in a one-by-one fashion. However, two vector pairs $\{p_1, p_2\}$ and $\{q_1, q_2\}$ can be linearly dependent although each vector is independent from each of the other three vectors.

We can test *one* of the strong Eigenvectors, say $p_1 \in \mathbf{V}_P \cdot \check{\mathbf{E}}_P$ whether or not it is linearly independent (in our relaxed sense) to *all* the strong Eigenvectors $q_i \in \mathbf{V}_Q \cdot \check{\mathbf{E}}_Q$ by substituting it into the local similarity matrix of Q, i.e. by testing: $p_1^{\mathbf{T}} \cdot \hat{\mathbf{M}}_Q \cdot p_1 > \Delta^2$.

If this comparison holds then we know that p_1 opens up a new dimension compared to Q, and that the correlation dimensionality $\lambda(P, Q)$ is at least $(\lambda_Q + 1)$. But if we test a second vector $p_2 \in \mathbf{V}_P \cdot \check{\mathbf{E}}_P$ we still have the problem that p_2 can be linearly dependent from $\mathbf{V}_Q \cdot \check{\mathbf{E}}_Q \bigcup \{p_1\}$ without being linearly dependent from any vector in $\mathbf{V}_Q \cdot \check{\mathbf{E}}_Q$ and $\{p_1\}$ alone. This problem is visualized in Figure 3. The strong Eigenvectors p_1 and p_2 of P (depicted in dashed lines) are each linearly independent from the strong Eigenvectors q_1 and q_2 of Q (depicted in solid lines) and by definition also linearly independent from each other (they are even orthogonal). However, p_2 is linearly dependent from the vectors in $\mathbf{V}_Q \cdot \check{\mathbf{E}}_Q \bigcup \{p_1\}$.

Proceedings of the 18th International Conference on Scientific and Statistical Database Management (SSDBM'06)
0-7695-2590-3/06 $20.00 © 2006 IEEE

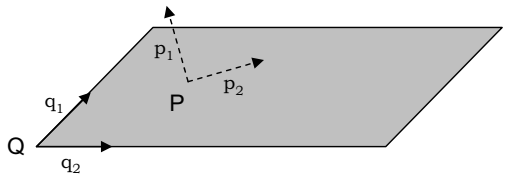


Figure 3. Two points with their Eigenvectors.

Therefore, before testing p_2, we have to integrate p_1 temporarily into the Eigenvector matrix $\mathbf{V}_Q$ (but only if p_1 indeed opens up a new dimension). To do so, we have to replace temporarily the *weak* Eigenvector q_{λ_Q+1} by the new *strong* Eigenvector p_1 and then orthonormalize the resulting matrix.

To orthonormalize the set of vectors $\{q_1, \ldots, q_{\lambda_Q}, p_1, q_{\lambda_Q+2}, \ldots, q_d\}$ the following steps have to be applied according to the method of Gram-Schmitt: Note that the vector q_{λ_Q+1} is temporarily replaced by vector p_1, i.e. $q_{\lambda_Q+1} = p_1$.

1. $x_i := q_i - \sum_{k=1}^{i-1} \langle q_k, q_i \rangle q_k$ for $i = \lambda_Q + 1, \ldots, d$
2. $q_i := \frac{x_i}{||x_i||}$ for $i = \lambda_Q + 1, \ldots, d$

The resulting vectors $\{q_1, \ldots, q_d\}$ build now again an orthonormal basis of the d-dimensional feature space.

We have to make $(d - \lambda_Q)$ vectors orthogonal which causes some problems because (1) we have to guarantee the linear independence (this time in the strong algebraic sense) of the remaining Eigenvectors in $\mathbf{V}_Q$ because, otherwise, orthonormalization could fail. (2) The effort is considerable high because this orthonormalization (which is in $O(d^2)$) must be performed every time a new vector is integrated into $\mathbf{V}_Q$. Therefore, our actual algorithm computes the test $p_i^{\mathbf{T}} \cdot (\mathbf{V}_Q \cdot \hat{\mathbf{E}}_Q \cdot \mathbf{V}_Q^{\mathbf{T}}) \cdot p_i > \Delta^2$ in an indirect way by replacing $\hat{\mathbf{E}}$ by $\check{\mathbf{E}}$ which will yield the advantage that less vectors (one instead of up to d vectors) have to be orthonormalized. The justification for the indirect computation is given by the following lemma:

Lemma 1 (Indirect Similarity Computation)
Let $\mathbf{V}$ *be an orthonormal matrix consisting of the strong Eigenvectors of* $\mathbf{\Sigma}_Q$*, some of the added and orthonormalized Eigenvectors of* $\mathbf{\Sigma}_P$ *and the remaining orthonormalized weak Eigenvectors of* $\mathbf{\Sigma}_Q$*. Than*

$$x^{\mathbf{T}} \cdot (\mathbf{V} \cdot \hat{\mathbf{E}} \cdot \mathbf{V}^{\mathbf{T}}) \cdot x = x^{\mathbf{T}} \cdot x - x^{\mathbf{T}} \cdot (\mathbf{V} \cdot \check{\mathbf{E}} \cdot \mathbf{V}^{\mathbf{T}}) \cdot x$$

Proof.

$$\begin{aligned} x^{\mathbf{T}} \cdot (\mathbf{V} \cdot \hat{\mathbf{E}} \cdot \mathbf{V}^{\mathbf{T}}) \cdot x = x^{\mathbf{T}} \cdot (\mathbf{V} \cdot (I - \check{\mathbf{E}}) \cdot \mathbf{V}^{\mathbf{T}}) \cdot x &= \\ x^{\mathbf{T}} \cdot (\mathbf{V} \cdot I \cdot \mathbf{V}^{\mathbf{T}}) \cdot x - x^{\mathbf{T}} \cdot (\mathbf{V} \cdot \check{\mathbf{E}} \cdot \mathbf{V}^{\mathbf{T}}) \cdot x &= \\ x^{\mathbf{T}} \cdot x - x^{\mathbf{T}} \cdot (\mathbf{V} \cdot \check{\mathbf{E}} \cdot \mathbf{V}^{\mathbf{T}}) \cdot x & \end{aligned}$$

The advantage of this computation is that now in the joint matrix $\check{\mathbf{M}}_Q = \mathbf{V}_Q \cdot \check{\mathbf{E}}_Q \cdot \mathbf{V}_Q^{\mathbf{T}}$ the weak Eigenvectors q_m for $m > \lambda_Q$ are not considered. Keeping the weak Eigenvectors orthonormal after every insertion of a new strong Eigenvector of $\mathbf{\Sigma}_P$ causes the main effort in orthonormalization: With direct computation, up to d vectors have to be orthonormalized after each insertion. Therefore, the overall complexity is $O(d^2)$ per insertion. Using the indirect computation it is sufficient to orthonormalize only the inserted vector which can be done in $O(d)$ time. Note also that in this case the linear independence of the vector to be orthonormalized to the strong Eigenvectors in $\mathbf{V}_Q$ is given, because this vector is even linear independent in our relaxed sense (and linear independency in weak sense implies linear independency in strict sense).

The algorithm for computing the correlation distance is presented in Figures 4 and 5. First, the correlation dimensionality $\lambda(P, Q)$ for a pair of points (P, Q) is derived as follows: For each of the *strong* Eigenvectors q_i of Q test whether $q_i^{\mathbf{T}} \cdot q_i - q_i^{\mathbf{T}} \cdot (\mathbf{V}_P \cdot \check{\mathbf{E}}_P \cdot \mathbf{V}_P^{\mathbf{T}}) \cdot q_i > \Delta^2$. If so, increase λ_P by one and set p_{λ_P} to the orthonormalized vector of q_i. Finally, $\lambda_P(Q)$ contains the correlation dimensionality of the point pair (P, Q) w.r.t. P. In an analogue way $\lambda_Q(P)$ is derived for the same point pair. The overall correlation dimensionality $\lambda(P, Q)$ is the maximum of both, $\lambda_P(Q)$, and $\lambda_Q(P)$. $\lambda(P, Q)$ is now the major building block for our correlation distance. As $\lambda(P, Q) \in \mathbb{N}$, many distances between different point pairs are identical. Therefore, there are many tie situations during clustering. We resolve these tie situations by additionally considering the Euclidean distance as a secondary criterion. This means, inside a correlation cluster (if there are no nested stronger correlation clusters), the points are clustered as by a conventional hierarchical clustering method. Formally we define:

Definition 6 (correlation distance)
The correlation distance *between two points* $P, Q \in \mathcal{D}$*, denoted by* $\text{CDIST}(P,Q)$*, is a pair consisting of the correlation dimensionality of* P *and* Q *and the Euclidean distance between* P *and* Q*, i.e.* $\text{CDIST}(P, Q) = (\lambda(P, Q), dist(P, Q))$.

We say $\text{CDIST}(P, Q) \leq \text{CDIST}(R, S)$ *if one of the following conditions hold:*

(1) $\lambda(P, Q) < \lambda(R, S)$,

(2) $\lambda(P, Q) = \lambda(R, S)$ *and* $dist(P, Q) \leq dist(R, S)$.

Proceedings of the 18th International Conference on Scientific and Statistical Database Management (SSDBM'06)
0-7695-2590-3/06 $20.00 © 2006 IEEE

function correlationDistance(P, Q, Δ)

compute $\check{\mathbf{E}}_P$ from $\hat{\mathbf{E}}_P$;
$\mathbf{V}_P$ = Eigenvector matrix of P;
λ_P = correlation dimensionality of P;

compute $\check{\mathbf{E}}_Q$ from $\hat{\mathbf{E}}_Q$;
$\mathbf{V}_Q$ = Eigenvector matrix of Q;
λ_Q = correlation dimensionality of Q;

for each strong Eigenvector $q_i \in \mathbf{V}_Q$ **do**
  if $q_i^{\mathbf{T}} q_i - q_i^{\mathbf{T}} \mathbf{V}_P \check{\mathbf{E}}_P \mathbf{V}_P^{\mathbf{T}} q_i > \Delta^2$ **then**
    adjust($\mathbf{V}_P$, $\check{\mathbf{E}}_P$, q_i, λ_P);
    $\lambda_P = \lambda_P + 1$;

for each strong Eigenvector $p_i \in \mathbf{V}_P$ **do**
  if $p_i^{\mathbf{T}} p_i - p_i^{\mathbf{T}} \mathbf{V}_Q \check{\mathbf{E}}_Q \mathbf{V}_Q^{\mathbf{T}} p_i > \Delta^2$ **then**
    adjust($\mathbf{V}_Q$, $\check{\mathbf{E}}_Q$, p_i, λ_Q);
    $\lambda_Q = \lambda_Q + 1$;

$\mathrm{CDIST} = \max(\lambda_P, \lambda_Q)$;
return (CDIST, $\mathrm{dist}_{\mathrm{Euclid}}(\mathrm{P}, \mathrm{Q})$);

Figure 4. Pseudocode correlation distance.

procedure adjust($\mathbf{V}$, $\check{\mathbf{E}}$, x, λ)

// set column $(\lambda + 1)$ of matrix V to vector x
$v_{\lambda+1} := x$;
for each strong Eigenvector $v_i \in V$ **do**
  $v_{\lambda+1} := v_{\lambda+1} - \langle v_i, x \rangle \cdot v_i$
$v_{\lambda+1} := \frac{v_{\lambda+1}}{||v_{\lambda+1}||}$;

set column $(\lambda+1)$ of $\check{\mathbf{E}}$ to the $(\lambda+1)$-th unit vector;

Figure 5. Pseudocode orthonormalization.

algorithm HiCO($\mathcal{D}$, k, μ, α, Δ)

for each $P \in \mathcal{D}$ **do**
  compute $\hat{\mathbf{E}}_P, \mathbf{V}_P$;

// priority queue pq is ordered by CREACH_μ
for each $P \in \mathcal{D}$ **do**
  $P.\mathrm{CREACH} = \infty$;
  insert P into pq;

while $pq \neq \emptyset$ **do**
  $O := pq$.next();
  $R :=$ μ-nearest neighbor if O;
  for each $P \in pq$ **do**
    new_cr := $\max(\mathrm{CDIST}(O, R), \mathrm{CDIST}(O, P))$;
    pq.update(P, new_cr);

Figure 6. Pseudocode HiCO algorithm.

4.2. Algorithm HiCO

Using the correlation distance defined above as a distance measure, we can basically run every hierarchical (or even non-hierarchical) clustering algorithm which is based on distance comparisons. Examples for such algorithms are Single-Link, Complete-Link, and the density-based clustering methods DBSCAN (non-hierarchical) and OPTICS. Since OPTICS is hierarchical and more robust w.r.t. noise than Single- and Complete-Link, we use the algorithmic schema and visualization technique of OPTICS for HiCO.

As suggested in [4] we introduce a smoothing factor μ to avoid the Single-Link effect and to achieve robustness against noise points. Thus, instead of using the correlation distance $\mathrm{CDIST}(P, Q)$ to measure the similarity of two points P and Q we use the *correlation reachability* $\mathrm{CREACH}_\mu(O, P)$ to compare these two points. The correlation reachability of a point P relative from a point O is defined as the maximum value of the correlation distance from O to its μ-nearest neighbor and the correlation distance between P and O.

Definition 7 (correlation reachability)
For $\mu \in \mathbb{N}$, $\mu \leq |\mathcal{D}|$ let R be the μ-nearest neighbor of $O \in \mathcal{D}$ w.r.t. the correlation distance. The correlation reachability *of a point $P \in \mathcal{D}$ relative from point O w.r.t. $\mu \in \mathbb{N}$ is defined as*

$$\mathrm{CREACH}_\mu(O, P) = \max(\mathrm{CDIST}(O, R), \mathrm{CDIST}(O, P))$$

Using this correlation reachability, HiCO computes a "walk" through the data set and assigns to each point O its smallest correlation reachability with respect to a point considered before O in the walk. In each step of the algorithm HiCO selects that point O having the minimum correlation reachability to any already processed point. This process is managed by a seed list which stores all points that have been reached from already processed points sorted according to the minimum correlation reachabilities. A special order of the database according to its correlation-based clustering structure is generated, the so-called cluster order, which can be displayed in a correlation reachability diagram. Such a correlation reachability diagram consists of the reachability values on the y-axis of all points, plotted in the order which HiCO produces on the x-axis. The result is a visualization of the clustering structure of the data set which is very easy to understand. The "valleys" in the plot represent the clusters, since points within a cluster typically have lower correlation reachabilities than points outside a cluster.

The complete integration of our distance measure into the algorithm HiCO can be seen in Figure 6.

4.3. Runtime Complexity

Let n be the number of data points and d be the dimensionality of the data space. In the preprocessing step the correlation neighborhoods are precomputed

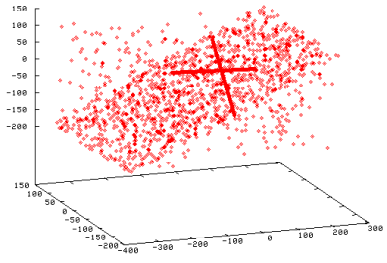


Figure 7. 3D synthetic data set (DS1).

which requires $O(nd+kd^2)$ for the determination of the covariance matrix. Since this is done for each object in the data set and $k < n$, we have a runtime complexity of $O(n^2d^2)$ for the preprocessing step. During the run of HiCO, we have to evaluate for each pair of points of the database its correlation dimensionality. This requires again a complexity of $O(n^2d^2)$. In addition, we have to decompose the covariance matrix of each point into the Eigenvalue matrix and the Eigenvector matrix. This step has a complexity of $O(nd^3)$. Thus, the overall runtime complexity of HiCO is $O(n^2d^2 + nd^3)$.

5. Experimental Evaluation

5.1. Data Sets

For our experiments, we used several synthetic data sets containing points marked with cluster labels that represent the hierarchical clustering structure. In addition, we used four real-world data sets. The first one, called "DS2", is a data set derived from a medical study to develop screening methods in order to identify carriers of a rare genetic disorder. Four measurements were made on blood samples of approximately 150 people who do not suffer from the disease and on 50 carriers of the disease.

As a second data set we used the "El Nino" data, a benchmark data set from the UCI KDD Archive[2]. The data set called "DS3" contains oceanographic and surface meteorological readings taken from a series of buoys positioned throughout the equatorial Pacific. It contains approximately 800 9D objects. The third data set called "DS4" consists of approximately 550 11D observations from the 1985 Current Population Survey[3]. The fourth data set called "DS5" consists of the concentrations of 43 metabolites in 2,000 newborns. The newborns were labeled according to some specific metabolic diseases.

[2] http://kdd.ics.uci.edu/

[3] http://lib.stat.cmu.edu/datasets/CPS_85_Wages

5.2. Results on Synthetic Data

We applied HiCO to several synthetic data sets. In the following, we focus on the 3-dimensional data set "DS1" depicted in Figure 7. It contains a hierarchy of correlation clusters consisting of two 1D correlations (lines) belonging to a 2D correlation (plane) and noise. The correlation reachability distance diagram computed by HiCO is shown in Figure 8(a). As it can be observed, HiCO detects two 1D correlation clusters that are embedded within a 2D correlation cluster. Additionally, some objects have a correlation distance of 3 (which equals the data dimensionality), i.e. they can be regarded as noise. We analyzed the "valleys" in the correlation reachability diagram marked with "Cluster 1", "Cluster 2", and "Cluster 3". The points that are clustered together in that correlation clusters are depicted in Figures 8(b), 8(c), and 8(d). As it can be seen, the correlation plane "Cluster 3" corresponds to the 2D correlation cluster in the diagram, whereas the two correlation lines "Cluster 1" and "Cluster 2" exactly correspond to the 1D correlation sub-clusters of "Cluster 3" in the diagram. Obviously, HiCO detects the hidden correlation hierarchy exactly.

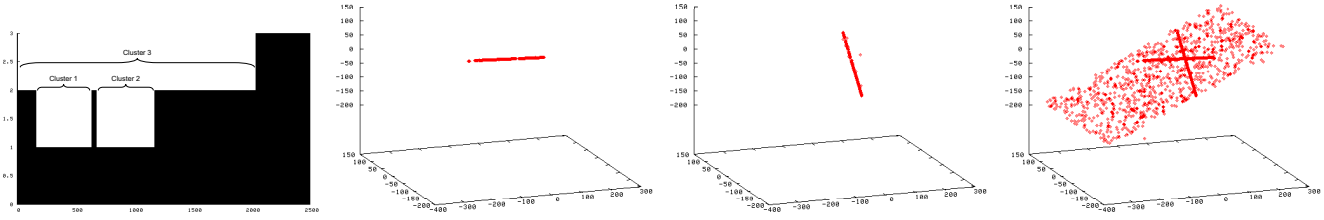
For comparison, we applied ORCLUS, OPTICS and 4C on the same datasets, but none of them were able to find the correlation clusters equally well, despite reasonable parameter settings. For ORCLUS we choose e.g. $k = 3$ and $l = 2$, but as it can be seen in Figure 9, ORCLUS was not able to find the correlation clusters hidden in the synthetic 3D data set.

We also applied OPTICS to the synthetic 3D data set (cf. Figure 10). OPTICS detected a hierarchy of clusters according to its density based paradigm, but it was not able to separate the correlation within these clusters.

Figure 11 shows the results of two 4C runs with different parameter settings. As parameter λ was set to one in the first run, 4C detected four 1-dimensional clusters consisting of the two lines in the synthetic 3D data set (cf. Figure 11(a)), but 4C failed to detect the 2-dimensional correlations. According to the parameter setting of $\lambda = 2$ in the second run, 4C found one 2-dimensional correlation cluster consisting of the two lines and the plane (cf. Figure 11(b)). In both runs, 4C was not able to detect all three correlation clusters as HiCO did.

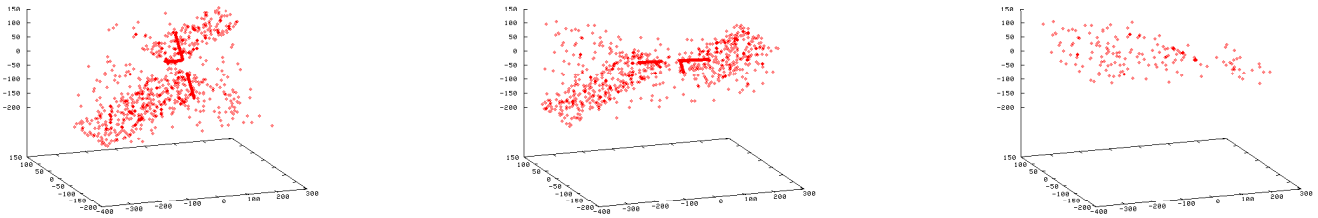
5.3. Real-world Data

The results of HiCO applied to the real-world data sets are shown in Figure 12. Applied to the DS2 data (cf. Figure 12(a)), HiCO found a cluster with cor-



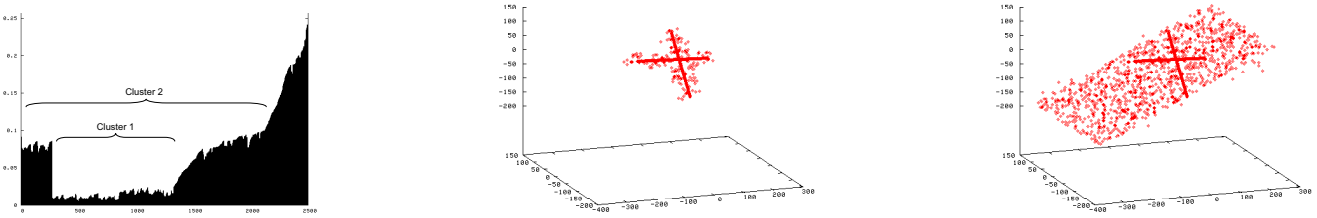
(a) Reachability plot. (b) Cluster 1. (c) Cluster 2. (d) Cluster 3.

Figure 8. Results of HiCO applied to DS1 (Parameters: $\mu = k = 20$, $\alpha = 0.90$, $\Delta = 0.05$).



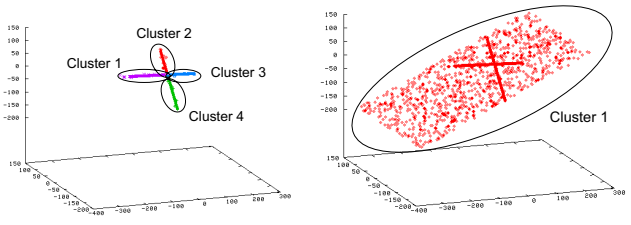
(a) Cluster 1. (b) Cluster 2. (c) Cluster 3.

Figure 9. Results of ORCLUS applied to DS1 (Parameters: $k = 3$, $l = 2$).



(a) Reachability plot. (b) Cluster 1. (c) Cluster 2.

Figure 10. Results of OPTICS applied to DS1 (Parameters: $\varepsilon = 1$, $minPts = 20$).



(a) Parameters: λ=1, ε=0.1, μ=20, δ=0.2. (b) Parameters: λ=2, ε=0.25, μ=20, δ=0.1.

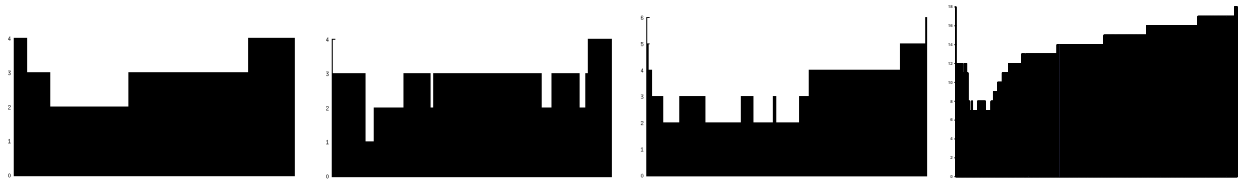
Figure 11. Results of 4C applied to DS1.

relation dimensionality of 2 embedded in a larger 3-dimensional correlation cluster. The cluster with a correlation dimensionality of 2 mostly consists of carriers of the genetic disorder. Most of the people not suffering from the disease belong to the cluster with a correlation dimensionality of 3.

The resulting reachability diagram of HiCO applied on data set DS3 is depicted in Figure 12(b). As it can be seen, the hierarchy contains a 1-dimensional correlation cluster and four 2-dimensional clusters. Analyzing these clusters, we found that the observations belonging to these clusters were mostly made from neighbored buoys.

The result of HiCO on DS4 is depicted in Figure 12(c). We can observe a strong hierarchy of correlation clusters. HiCO computed four 2-dimensional correlation clusters embedded in a 3-dimensional correlation cluster which is again embedded in a 4-dimensional cluster. The hierarchy ends up with 5-dimensional and 6-dimensional clusters. The first of the 2-dimensional

Proceedings of the 18th International Conference on Scientific and Statistical Database Management (SSDBM'06)
0-7695-2590-3/06 $20.00 © 2006 IEEE



(a) DS2 (μ=10, k=25). (b) DS3 (μ=15, k=40). (c) DS4 (μ=10, k=40). (d) DS5 (μ=10, k=100).

Figure 12. Results of HiCo on real-world data sets (Parameters: $\Delta = 0.25$, $\alpha = 0.8$).

clusters consists only of white married women, living not in the southern states of the USA and not belonging to any union. To the second 2-dimensional cluster male persons with the same attributes as the women in the first cluster have been assigned. The third 2-dimensional cluster consists of unmarried white women being no union member and living in the northern states. And last but not least people belonging to the fourth 2-dimensional cluster have the same attributes as the third cluster but being men instead of women. Obviously, HiCO computed pure correlation clusters on this data set.

Finally, HiCO retrieved on DS5 7-dimensional and 8-dimensional correlation clusters embedded in higher dimensional clusters (cf. Figure 12(d)). These clusters of relative low dimensionality consist only of newborns suffering from phenylketonuria (PKU), while the healthy newborns are grouped in the clusters of higher dimensionality.

To summarize, our experiments show that HiCO detects several interesting correlation cluster hierarchies in real-world data sets.

6. Conclusions

In this paper, we have proposed HiCO, the first algorithm for computing hierarchies of correlation clusters, i.e. lower dimensional correlations embedded within larger dimensional correlation clusters. In contrast to existing approaches, our method does not require the user to specify any global density threshold, the number of clusters to be found, nor any parameter specifying the dimensionality of the correlations. Our extensive experimental evaluation shows that HiCO finds meaningful and rich hierarchies of correlation clusters in synthetic and real-world data sets.

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. SIGMOD*, 1999.

[2] C. C. Aggarwal and P. S. Yu. Finding generalized projected clusters in high dimensional space. In *Proc. SIGMOD*, 2000.

[3] R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan. Automatic subspace clustering of high dimensional data for data mining applications. In *Proc. SIGMOD*, 1998.

[4] M. Ankerst, M. M. Breunig, H.-P. Kriegel, and J. Sander. OPTICS: Ordering points to identify the clustering structure. In *Proc. SIGMOD*, 1999.

[5] C. Böhm, K. Kailing, H.-P. Kriegel, and P. Kröger. Density connected clustering with local subspace preferences. In *Proc. ICDM*, 2004.

[6] C. Böhm, K. Kailing, P. Kröger, and A. Zimek. Computing clusters of correlation connected objects. In *Proc. SIGMOD*, 2004.

[7] Y. Cheng and G. M. Church. Biclustering of expression data. In *Proc. ISMB*, 2000.

[8] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proc. KDD*, 1996.

[9] J. A. Hartigan. Direct clustering of a data matrix. *JASA*, 67(337):123–129, 1972.

[10] K. Kailing, H.-P. Kriegel, and P. Kröger. Density-connected subspace clustering for high-dimensional data. In *Proc. SDM*, 2004.

[11] J. Liu and W. Wang. OP-Cluster: Clustering by tendency in high dimensional spaces. In *Proc. ICDM*, 2003.

[12] J. Pei, X. Zhang, M. Cho, H. Wang, and P. S. Yu. MaPle: A fast algorithm for maximal pattern-based clustering. In *Proc. ICDM*, 2003.

[13] C. M. Procopiuc, M. Jones, P. K. Agarwal, and T. M. Murali. A Monte Carlo algorithm for fast projective clustering. In *Proc. SIGMOD*, 2002.

[14] A. K. H. Tung, X. Xu, and C. B. Ooi. CURLER: Finding and visualizing nonlinear correlated clusters. In *Proc. SIGMOD*, 2005.

[15] H. Wang, W. Wang, J. Yang, and P. S. Yu. Clustering by pattern similarity in large data sets. In *Proc. SIGMOD*, 2002.

[16] J. Yang, W. Wang, H. Wang, and P. S. Yu. Delta-Clusters: Capturing subspace correlation in a large data set. In *Proc. ICDE*, 2002.