On Exploring Complex Relationships of Correlation Clusters

Elke Achtert, Christian Böhm, Hans-Peter Kriegel, Peer Kröger, Arthur Zimek
Institute for Informatics, Ludwig-Maximilians-Universität München, Germany
`http://www.dbs.ifi.lmu.de`

E-mail: {`achtert,boehm,kriegel,kroegerp,zimek`}`@dbs.ifi.lmu.de`

Abstract

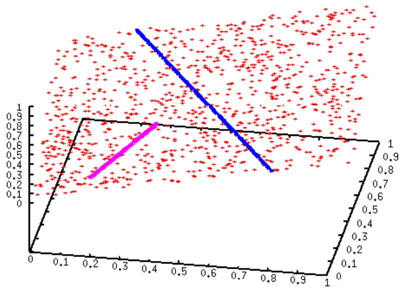
In high dimensional data, clusters often only exist in arbitrarily oriented subspaces of the feature space. In addition, these so-called correlation clusters may have complex relationships between each other. For example, a correlation cluster in a 1-D subspace (forming a line) may be enclosed within one or even several correlation clusters in 2-D superspaces (forming planes). In general, such relationships can be seen as a complex hierarchy that allows multiple inclusions, i.e. clusters may be embedded in several super-clusters rather than only in one. Obviously, uncovering the hierarchical relationships between the detected correlation clusters is an important information gain. Since existing approaches cannot detect such complex hierarchical relationships among correlation clusters, we propose the algorithm ERiC to tackle this problem and to visualize the result by means of a graph-based representation. In our experimental evaluation, we show that ERiC finds more information than state-of-the-art correlation clustering methods and outperforms existing competitors in terms of efficiency.

1. Introduction

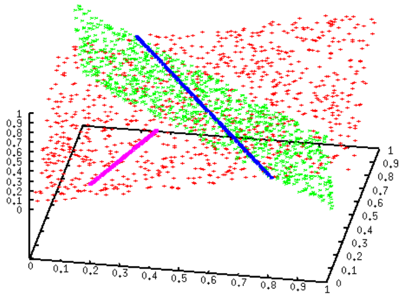
In high-dimensional data, meaningful clusters are usually based only on a subset of all dimensions. Subspace clustering (or projected clustering) is a well known approach to find λ-dimensional subspaces of a d-dimensional data space ($\lambda < d$), where certain sets of points cluster well. Despite great efforts in developing subspace clustering methods, all existing approaches suffer from certain drawbacks. Each approach is based on certain heuristics because the optimal solution would require at least a time complexity exponential in the number of dimensions d. An even more challenging problem is to find clusters in arbitrarily oriented subspaces. Such clusters appear as sets of points located near a common hyperplane (of arbitrary dimension λ) in a d-dimensional data space. Since these hyperplanes correspond to linear dependencies among several attributes (and thus the corresponding attributes are correlated), the concept of knowledge discovery in databases addressing this problem is known as correlation clustering [10]. Correlation clustering groups the data sets into subsets called correlation clusters such that the objects in the same correlation cluster are all associated to a common hyperplane of arbitrary dimensionality. A correlation cluster associated to a λ-dimensional hyperplane is referred to as a *λ-dimensional correlation cluster*. The dimensionality of a hyperplane associated to a correlation cluster is called the *correlation dimensionality*.

A good example of a successful application of correlation clustering are recommendation systems. In target marketing, it is important to find homogeneous groups of users with similar ratings in subsets of the attributes. In addition, it is interesting to find groups of users with correlated affinities. This knowledge can help companies to predict customer behavior and thus develop future marketing plans. A second application of correlation clustering is metabolic screening. The collected data usually contain the concentrations of certain metabolites in blood samples of thousands of patients. In such data sets, it is important to find homogeneous groups of patients with correlated metabolite concentrations indicating a common metabolic disease. This is an important step towards understanding metabolic or genetic disorders and designing individual drugs. Another prominent application for correlation clustering is the analysis of gene expression data. Gene expression data contain the expression levels of thousands of genes, indicating how *active* the genes are, according to a set of samples. A common task is to find clusters of co-regulated genes, i.e. clusters of genes that share a common linear dependency within a set of their features.

The first approach that can detect correlation clusters is ORCLUS [6] that integrates PCA into k-means clustering. The algorithm 4C [10] integrates PCA into a density-based clustering algorithm. These approaches can be seen as "flat" approaches in the following sense. A correlation cluster C_1 with dimensionality λ_1 may be embedded in (and therefore



(a) Sample data set 1



(b) Sample data set 2

Figure 1. Simple (a) and complex (b) hierarchical relationships among correlation clusters

may be part of) another correlation cluster C_2 with dimensionality $\lambda_2 > \lambda_1$. In general, there may be a kind of hierarchy among correlation clusters that are embedded into higher dimensional correlation clusters. Since ORCLUS and 4C cannot detect such hierarchies, the algorithm HiCO [4] was proposed tackling correlation clustering as a hierarchical problem, i.e. exploring the information of correlation clusters of lower correlation dimensionality that together form a larger correlation cluster of higher correlation dimensionality. Although it is represented by the same models (dendrogram/reachability diagram), the resulting hierarchy is different from the hierarchies computed by traditional hierarchical clustering algorithms such as Single-Link or OPTICS [8]. The hierarchy among correlation clusters reflects the relationships among the subspaces in which these correlation clusters are embedded rather than spatial vicinity or density. As a simple illustration consider the data set depicted in Figure 1(a): Two lines, i.e. 1-D correlation clusters, are embedded within a plane, i.e. a 2-D correlation cluster. The resulting hierarchy consists of the two 1-D clusters as leaf-nodes of the hierarchy-tree both having the 2-D correlation cluster as parent node. HiCO aims at generating a tree-based representation of the correlation cluster hierarchy.

However, it may not always be appropriate to reflect the hierarchy of correlation clusters as a tree. A correlation cluster may be embedded in several correlation clusters of higher dimensionality, resulting in a hierarchy with *multiple inclusions* (similar to the concept of "multiple inheritance" in software engineering). Consider e.g. the data set depicted in Figure 1(b): One of the 1-D correlation clusters is the intersection of two 2-D correlation clusters, i.e. it is embedded within two clusters of higher dimensionality. Those multiple inclusions can only be represented by a graph-based visualization approach which is beyond the capabilities of previous methods such as HiCO.

In this paper, we propose a new algorithm called ERiC (Exploring Relationships among Correlation clusters) to completely uncover any complex hierarchical relationships of correlation clusters in high dimensional data sets including not only single inclusions (like HiCO) but also multiple inclusions. In addition, ERiC provides a clear visualization of these complex relationships by means of a graph-based representation.

The remainder of the paper is organized as follows. We discuss the related work in Section 2. In Section 3, we describe the notion of correlation clusters based on PCA in more detail in preparation to handle correlation clusters formally. Section 4 describes our new approach. An experimental evaluation is presented in Section 5. Section 6 concludes the paper.

2. Related Work

In recent years, several subspace clustering algorithms have been proposed [7, 5, 19, 16, 9] to uncover clusters in axis parallel projections of the original data set. Some approaches even provide information regarding the hierarchical relationships among different subspace clusters [1, 2]. However, subspace clustering algorithms are not able to capture local data correlations and find clusters of correlated objects since the principal axes of correlated data are arbitrarily oriented.

Pattern-based clustering methods [21, 22, 17, 18] aim at grouping objects that exhibit a similar trend in a subset of attributes into clusters rather than grouping objects with low distance. This problem is also known as co-clustering or biclustering [15]. In contrast to correlation clustering, pattern-based clustering limits itself to a very special form of correlation where all attributes are positively correlated. It does not include negative correlations or correlations where one attribute is determined by a linear combination of two or more other attributes. Thus, biclustering or pattern-based clustering could be regarded as a special case of correlation clustering, as more extensively discussed in [10].

The expectation maximization (EM) algorithm [12] is one of the first clustering algorithms that can generally detect correlation clusters. The EM algorithm tries to model the data distribution of a data set using a mixture of non-axis parallel Gaussian distributions. However, the EM algorithm cannot distinguish between correlation clusters and full-dimensional clusters without any correlation. In addition, it favors clusters of spherical shape and requires the user to specify the number of clusters in advance. As a main drawback, the EM clustering is very sensitive to noise.

ORCLUS [6] is a k-means style correlation clustering algorithm and, thus, can be seen as a specialization of EM that detects only correlation clusters. The correlation clusters are allowed to exist in arbitrarily oriented subspaces represented by a set of Eigenvectors. Like most k-means based approaches, ORCLUS favors correlation clusters of spherical shape and requires the user to specify the number of clusters in advance. If the user's guess does not correspond to the actual number of clusters, the results of ORCLUS deteriorate considerably. A second problematic parameter of ORCLUS is the dimensionality l of the correlation clusters ORCLUS is desired to uncover. ORCLUS usually has problems with correlation clusters of very different dimensionalities because the resulting clusters must have dimensionalities near l. Furthermore, ORCLUS is in general very sensitive to noise similar to EM and its derivatives. A very similar method is presented in [11] for the purpose of enhancing indexing of high dimensional data.

In [10] the algorithm 4C is presented to find clusters of correlation-connected objects. 4C is a combination of DBSCAN [13] and PCA and searches for arbitrary linear correlations of fixed dimensionality. The user must specify several parameters, including: ε and μ, defining minimum density of a cluster, a threshold δ to decide which principal axes of a cluster are relevant for the correlation, and the dimensionality λ of the computed correlation clusters. 4C may also miss important clusters for similar reasons as ORCLUS. A variation of 4C is the algorithm COPAC [3] that uses a different similarity measure.

In [20] the method CURLER to detect arbitrary, non-linear correlations has been proposed. CURLER uses the concept of micro-clusters that are generated using an EM variant and then are merged to discover correlation clusters. CURLER improves over ORCLUS and 4C as the correlations underlying the clusters are not necessarily linear. Furthermore, as a fuzzy approach, CURLER assumes each data object to belong to all clusters simultaneously, but with different probabilities for each cluster assigned. By merging several clusters according to their co-sharing level, the algorithm on the one hand becomes less sensitive to the pre-defined number k of clusters, thus also overcoming a severe limitation of any k-means related approach. On the other hand, the user cannot directly derive a model describing the correlations, since the original k models are no longer persistent in the resulting clustering. However, we focus on strong, linear correlations between features. Thus, the non-linear correlations discovered by CURLER are of no interest for our approach.

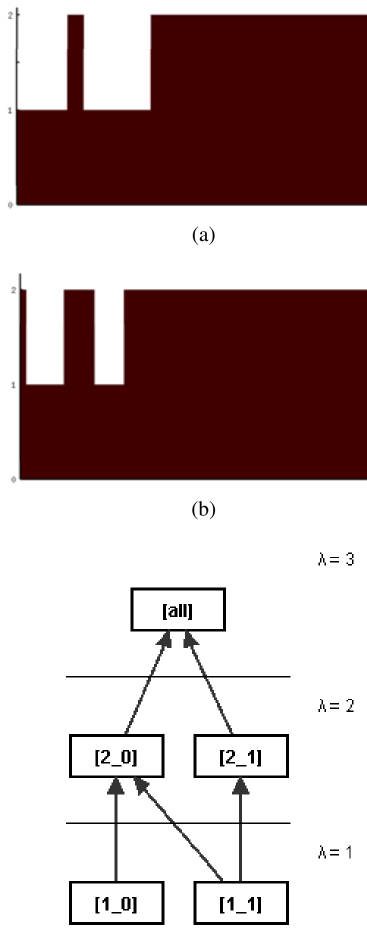


Figure 2. Results of HiCO on the data sets shown in Figure 1

Recently, the method DIC [14] has been proposed. DIC uses the concept of the fractal dimension in order to measure and model the correlation within the clusters. In particular, DIC represents each data object as a tuple containing the fractal dimension and the local density of the given object. Afterwards, the EM algorithm is applied to a data set containing these tuples of each data object. As a consequence, DIC suffers from the problem that the number of clusters must be estimated in advance and specified as an input parameter. In addition, DIC does not distinguish between usual (full-dimensional) clusters and correlation clusters that form a hyperplane in the data space.

The first and to our knowledge only approach deriving information regarding the hierarchical relationships among correlation clusters is HiCO [4]. HiCO incorporates a distance measure taking into account local correlation dimensionalities into the hierarchical clustering algorithm OPTICS [8]. The resulting reachability-plot allows to derive a simple hierarchy of correlation clusters. Let us consider two main drawbacks of HiCO: Firstly, HiCO uses a relatively complex distance measure for every distance query in the clustering step. This results in considerable computational efforts. Secondly, HiCO assumes a relatively simple hierarchy of correlation clusters. Multiple inclusions cannot be derived from the resulting plot. Thus, the detected hierarchical structure of correlation clusters can be misleading or even simply wrong. This limitation is illustrated in Figure 2 depicting the resulting reachability plot when applying HiCO on the sample datasets from Figure 1. As it can be observed, the resulting plots look almost identical for both, sample data set 1 (cf. Figure 2(a)) and sample data set 2 (cf. Figure 2(b)). Since valleys in the plot indicate clusters, both plots reveal the same information of two 1-D clusters embedded within one 2-D cluster. In fact, in data set 2 the two 2-D clusters cannot be separated and the complex hierarchy consisting of the multiple inclusion cannot be detected by HiCO. The true hierarchy hidden in sample data set 2 can only be represented by a graph model. Figure 2(c) envisions such a visualization of the complete hierarchy allowing for multiple inclusions. In fact, our method ERiC will produce such a visualization.

To summarize, we find HiCO as only competitor able to detect hierarchical structures of correlation clusters (albeit simplified by excluding multiple inclusions). As a baseline, ORCLUS and 4C can be of use for comparison with general correlation clustering methods without providing information w.r.t. hierarchical relationships among correlation clusters.

3. A Notion of Correlation Clusters

In this section, we prepare the introduction of our approach by formalizing the notion of correlation clusters. In the following, we assume $\mathcal{D}$ to be a database of n feature vectors in a d-dimensional feature space, i.e. $\mathcal{D} \subseteq \mathbb{R}^d$. A correlation cluster is a set of feature vectors that are close to a common, arbitrarily oriented subspace of a given dimensionality λ ($1 \leq \lambda < d$). In the data space, the correlation cluster appears as a hyperplane of dimensionality λ.

In general, one way to formalize the concept of correlation clusters is to use PCA. Formally, let $\mathcal{C}$ be a correlation cluster, i.e. $\mathcal{C} \subseteq \mathcal{D}$, and let $\bar{X}$ denote the centroid of all points in $\mathcal{C}$. The *$d \times d$ covariance matrix $\Sigma_\mathcal{C}$* of $\mathcal{C}$ is defined as:

$$\Sigma_\mathcal{C} = \frac{1}{|\mathcal{C}|} \cdot \sum_{X \in \mathcal{C}} (X - \bar{X}) \cdot (X - \bar{X})^\mathsf{T}.$$

Since the covariance matrix $\Sigma_\mathcal{C}$ of $\mathcal{C}$ is a positive semi-definite square matrix, it can be decomposed into the *eigenvalue matrix $E_\mathcal{C}$* of $\Sigma_\mathcal{C}$ and the *eigenvector matrix $V_\mathcal{C}$* of $\Sigma_\mathcal{C}$ such that $\Sigma_\mathcal{C} = V_\mathcal{C} \cdot E_\mathcal{C} \cdot V_\mathcal{C}^\mathsf{T}$. The eigenvalue matrix $E_\mathcal{C}$ is a diagonal matrix storing the d non-negative eigenvalues of $\Sigma_\mathcal{C}$ in decreasing order. The eigenvector matrix $V_\mathcal{C}$ is an orthonormal matrix with the corresponding d eigenvectors of $\Sigma_\mathcal{C}$.

Now we define the correlation dimensionality of $\mathcal{C}$ as the number of dimensions of the (arbitrarily oriented) subspace which is spanned by the major axes in $V_\mathcal{C}$. Let us note that the correlation dimensionality is closely related to the intrinsic dimensionality of the data distribution. If, for instance, the points in $\mathcal{C}$ 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 (eigenvectors) of $\Sigma_\mathcal{C}$. 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 $\Sigma_\mathcal{C}$ which is the total variance of the points in $\mathcal{C}$. 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 set of points $\mathcal{C}$ is now defined as the smallest number of eigenvectors explaining a portion of at least $\alpha \in]0, 1[$ of the total variance of $\mathcal{C}$.

In the following, we call the $\lambda_\mathcal{C}$-dimensional subspace which is spanned by the major axes of $\mathcal{C}$ the *correlation hyperplane* of $\mathcal{C}$. Since we follow the convention that the eigenvalues are ordered decreasingly in the eigenvalue matrix $E_\mathcal{C}$, the major axes correspond to the $\lambda_\mathcal{C}$ first eigenvectors of $\Sigma_\mathcal{C}$.

Thus, the correlation dimensionality $\lambda_\mathcal{C}$ is the dimensionality of the subspace containing all points of the set $\mathcal{C}$ allowing a small deviation corresponding to the remaining portion of variance of $1 - \alpha$. The remaining, neglected variance scatters along the eigenvectors $v_{\lambda_\mathcal{C}+1}, \ldots, v_d$.

4. Algorithm ERiC

As discussed above, hierarchical clustering schemata such as the agglomerative schema (e.g. used by Single-Link), the divisive schema, or the density-based schema (e.g. used by OPTICS) cannot uncover complex hierarchies that exhibit multiple inclusions. The reason for this is that the resulting complex hierarchy of an algorithm implementing any of the traditional schemata is only capable of producing a tree-like hierarchy rather than producing a graph-like hierarchy. Thus, approaches like HiCO, that integrate a suitable "correlation distance measure" into traditional hierarchical clustering schemata cannot be used to handle hierarchies with multiple inclusions.

As a consequence, ERiC follows a different strategy. The basic idea of ERiC is to first generate all correlation clusters and, second, to determine the hierarchy from this result. Obviously, during the computation of the clusters it would be very helpful to aggregate information that can be used to explore the hierarchical relationships among these clusters. In addition, it is required to compute all correlation clusters for all possible correlation dimensions simultaneously.

Since none of the existing correlation clustering algorithms meets both requirements we propose a novel approach to determine the complete set of correlation clusters and additional information for the hierarchy generation process. In particular, our algorithm ERiC consists of the following three steps: First, the objects of the database are partitioned w.r.t. their "correlation dimensionality" (cf. Section 4.1). This correlation dimensionality of a point $p \in \mathcal{D}$ will reflect the dimensionality of the correlation cluster in which p fits best. In a second step, the points within each partition are clustered using a "flat" correlation clustering algorithm (cf. Section 4.2). The result of these two steps is the complete set of correlation clusters with the additional information regarding their dimensionality. To explore the relationships among the correlation clusters found during step 2, we follow a bottom-up strategy. For any cluster $\mathcal{C}_i$ with correlation dimensionality λ_i, we consider those clusters $\mathcal{C}_j$ with correlation dimensionality $\lambda_j > \lambda_i$ as possible parents. A cluster $\mathcal{C}_j$ is a parent of $\mathcal{C}_i$, if $\mathcal{C}_i$ is embedded in (and therefore part of) $\mathcal{C}_j$. Using this information, ERiC creates the final result (i.e. a hierarchy of correlation clusters with multiple inclusions) in the third step (cf. Section 4.3).

4.1. Partitioning w.r.t. Correlation Dimensionality

In the first step of ERiC, we partition the database according to the *local correlation dimensionality* of the database objects reflecting the correlation dimensionality of the local neighborhood of each point.

Definition 1 (local correlation dimensionality)
Let $\alpha \in]0,1[$, $p \in \mathcal{D}$, and let $\mathcal{N}_p$ denote the set of points in the local neighborhood of p. Then the local correlation dimensionality λ_p *of the point p is the smallest number of eigenvalues e_i in the eigenvalue matrix $E_{\mathcal{N}_p}$ explaining a portion of at least α of the total variance, i.e.*

$$\lambda_p = \min_{r \in \{1,\ldots,d\}} \left\{ r \;\middle|\; \frac{\sum_{i=1}^{r} e_i}{\sum_{i=1}^{d} e_i} \geq \alpha \right\}$$

Let us note that $E_{\mathcal{N}_p}$ is the eigenvalue matrix of $\Sigma_{\mathcal{N}_p}$ which is the covariance matrix of $\mathcal{N}_p$. Typically, values for α are chosen between 0.8 and 0.9. For example, $\alpha = 0.85$ denotes that the obtained principal components explain 85% of the total variance. The set of points $\mathcal{N}_p$ of p should well reflect the correlation in the local neighborhood of p. Thus, one may e.g. choose the k-nearest neighbors of p as the neighborhood $\mathcal{N}_p$ of p. This way, one can ensure to consider a set of points large enough to derive a meaningful covariance matrix $\Sigma_{\mathcal{N}_p}$. Obviously, k should considerably exceed d. On the other hand, k should not be too large, as otherwise too many noise points may influence the derivation of the local correlation structure.

Based on Definition 1, the first step of ERiC partitions the database objects according to their local correlation dimensionality, derived from the k-nearest neighbors of each object. A point $p \in \mathcal{D}$ with $\lambda_p = i$ is assigned to a partition $\mathcal{D}_i$ of the database $\mathcal{D}$. This results in a set of d disjoint subsets $\mathcal{D}_1, \ldots, \mathcal{D}_d$ of $\mathcal{D}$. Some of these subsets may remain empty. In terms of correlation clustering, $\mathcal{D}_d$ contains noise, since there is no linear dependency of features within the neighborhood of p, if $\lambda_p = d$.

This first step of ERiC yields an appropriate correlation dimensionality for each point in advance. Furthermore, the number n of data points to process in the clustering step for each partition is reduced to $\frac{n}{d}$ on the average.

4.2. Computing Correlation Clusters within each Partition

Having performed the partitioning of the database $\mathcal{D}$ in step 1, a clustering step is performed for each partition separately. For the clustering procedure, we can utilize the fact that all points within a given partition $\mathcal{D}_i$ share a common local correlation dimensionality i. Based on the local correlation dimensionality of a point p, we distinguish *strong eigenvectors* that span the hyperplane associated with a possible correlation cluster containing p, and *weak eigenvectors* that are perpendicular to this hyperplane.

Definition 2 (strong and weak eigenvectors)
Let $p \in \mathcal{D}$, λ_p be the local correlation dimensionality of p, and let V_p be the corresponding eigenvectors of the point p based on the local neighborhood $\mathcal{N}_p$ of p. We call the first λ_p eigenvectors of V_p strong eigenvectors, *the remaining eigenvectors are called* weak.

To easily describe some matrix computations in the following, we define a selection matrix for weak eigenvectors as follows.

Definition 3 (selection matrix for weak eigenvectors)
Let $p \in \mathcal{D}$, λ_p be the local correlation dimensionality of p, and let E_p be the corresponding eigenvectors and eigenvalues of the point p based on the local neighborhood $\mathcal{N}_p$ of p. The selection matrix $\hat{E}_p$ for weak eigenvectors *with entries $\hat{e}_{ij} \in \{0,1\}$, $i,j = 1,\ldots,d$, is constructed according to the following rule:*

$$\hat{e}_{ij} = \begin{cases} 1 & \text{if} \quad i = j > \lambda_p \\ 0 & \text{otherwise} \end{cases}$$

Based on this definition, the *weak eigenvectors* of p are given by $V_p \cdot \hat{E}_p$.

For the clustering, we will associate two points, $p, q \in \mathcal{D}_i$, to the same cluster, if their strong eigenvectors span approximately the same hyperplane. This will not be the case, if any strong eigenvector of p is linearly independent from the strong eigenvectors of q or *vice versa*. The number i of strong eigenvectors is the same for p and q as both are placed in the same partition $\mathcal{D}_i$. But we can even define this condition more general for a different number of strong eigenvectors. However, we need to consider linear dependency in a weakened sense to allow a certain degree, say Δ, of deviation. In real world data, it is unlikely to find a correlation cluster that perfectly fits to a hyperplane. We therefore define an *approximate linear dependency* among the strong eigenvectors of two points.

Definition 4 (approximate linear dependency)
Let $\Delta \in]0, 1[$, $p, q \in \mathcal{D}$, and w.l.o.g. $\lambda_p \leq \lambda_q$. Then the strong eigenvectors of p are approximately linear dependent *from the strong eigenvectors of q if the following condition holds for all strong eigenvectors v_i of p:*

$$\sqrt{v_i^{\mathsf{T}} \cdot V_q \cdot \hat{E}_q \cdot V_q^{\mathsf{T}} \cdot v_i} \leq \Delta$$

If the strong eigenvectors of p are approximately linear dependent *from the strong eigenvectors of q, we write*

$$\mathrm{SPAN}(p) \subseteq_{\text{aff}}^{\Delta} \mathrm{SPAN}(q)$$

As indicated above, Δ specifies the degree of deviation of a straight plane a correlation cluster may exhibit.

Definition 4 does not take into account any affinity. Thus, we consider the strong eigenvectors of p approximately linear dependent from the strong eigenvectors of q ($\mathrm{SPAN}(p) \subseteq_{\text{aff}}^{\Delta} \mathrm{SPAN}(q)$), although possibly $p \notin \mathrm{SPAN}(q)$, i.e., the space spanned by the strong eigenvectors of p is (approximately) parallel to the space spanned by the strong eigenvectors of q. To exclude affine subspaces, we additionally assess the distance between p and q along the weak eigenvectors of q, i.e., perpendicular to the hyperplane defined by the strong eigenvectors of q. This distance which we call *affine distance* is defined as follows.

Definition 5 (affine distance)
Let $p, q \in \mathcal{D}$, w.l.o.g. $\lambda_p \leq \lambda_q$, and $\mathrm{SPAN}(p) \subseteq_{\text{aff}}^{\Delta} \mathrm{SPAN}(q)$. The affine distance *between p and q is given by*

$$\mathrm{DIST}_{\text{aff}}(p, q) = \sqrt{(p-q)^{\mathsf{T}} \cdot V_q \cdot \hat{E}_q \cdot V_q^{\mathsf{T}} \cdot (p-q)}$$

Combining approximate linear dependency (Definition 4) and the affine distance (Definition 5) yields the definition of a correlation distance between two points.

Definition 6 (correlation distance)
Let $\delta \in \mathbb{R}_0^+$, $\Delta \in]0, 1[$, $p, q \in \mathcal{D}$, and w.l.o.g. $\lambda_p \leq \lambda_q$. Then the correlation distance $\mathrm{CORRDIST}_{\Delta}^{\delta}$ *between two points $p, q \in \mathcal{D}$, denoted by $\mathrm{CORRDIST}_{\Delta}^{\delta}(p, q)$, is defined as follows*

$$\mathrm{CORRDIST}_{\Delta}^{\delta}(p, q) = \begin{cases} 0 & \textit{if } \mathrm{SPAN}(p) \subseteq_{\text{aff}}^{\Delta} \mathrm{SPAN}(q) \\ & \quad \wedge \mathrm{DIST}_{\text{aff}}(p, q) \leq \delta \\ 1 & \textit{otherwise} \end{cases}$$

The parameter δ specifies a certain degree of jitter. Two parallel subspaces M, N are considered distinct, if the affine distances $\mathrm{DIST}_{\text{aff}}(m, n)$ and $\mathrm{DIST}_{\text{aff}}(n, m)$ exceed δ for any two points $m \in M$ and $n \in N$, otherwise the subspaces are considered to be equal. Since the relations $\mathrm{SPAN}(p) \subseteq_{\text{aff}}^{\Delta} \mathrm{SPAN}(q)$ and $\mathrm{DIST}_{\text{aff}}(p, q)$ are based on the local neighborhood of q, they are not symmetric. For $\lambda_p < \lambda_q$, these measurements yield the notion of a subspace $\mathrm{SPAN}(p)$ embedded in another subspace $\mathrm{SPAN}(q)$ of higher dimensionality as required to deduct a hierarchy of subspaces. However, as a distance function for clustering within one partition, all clusters are supposed to exhibit equal dimensionality. We therefore construct a symmetric distance function as

$$\mathrm{dist}(\mathrm{p}, \mathrm{q}) = \max\left(\mathrm{CORRDIST}_{\Delta}^{\delta}(\mathrm{p}, \mathrm{q}), \mathrm{CORRDIST}_{\Delta}^{\delta}(\mathrm{q}, \mathrm{p})\right).$$

In each partition, we perform a density-based clustering using DBSCAN with dist as distance function. DBSCAN is chosen due to its efficiency, effectivity, and usability: The input parameter ε determining the range of neighborhood is set to 0 since the distance d is binary. The parameter `minPts` determines the minimum size of a cluster. This parameter can be intuitively set according to the nature of a given problem. As a result, we get a set of clusters for each partition $\mathcal{D}_i$.

4.3. Aggregating the Hierarchy of Correlation Clusters

As mentioned above, the parent of a cluster $\mathcal{C}_i$ with correlation dimensionality λ_i can be any cluster $\mathcal{C}_j$ with correlation dimensionality $\lambda_j > \lambda_i$. Each cluster $\mathcal{C}_i$ derived in step 2 gets assigned a centroid x_i as mean value over all cluster members. Then the cluster centroid x_i gets assigned a set of strong and weak eigenvectors using all cluster members as neighborhood $\mathcal{N}_{x_i}$ as specified in Definitions 1 and 3. Assuming the clusters sorted in ascending order w.r.t. their correlation dimensionality (as already given by the partitions $\mathcal{D}_1, \ldots, \mathcal{D}_d$), ERiC starts with the first cluster $\mathcal{C}_m$ and checks for each cluster $\mathcal{C}_n$ with $\lambda_n > \lambda_m$ whether $\mathrm{SPAN}(x_m) \subseteq_{\text{aff}}^{\Delta} \mathrm{SPAN}(x_n)$ and $\mathrm{DIST}_{\text{aff}}(x_m, x_n) \leq \delta$ according to Definitions 4–5, i.e. the $\mathrm{CORRDIST}_{\Delta}^{\delta}(x_m, x_n)$ is derived (Definition 6). If $\mathrm{CORRDIST}_{\Delta}^{\delta}(x_m, x_n) = 0$, cluster $\mathcal{C}_n$ is treated as parent of cluster $\mathcal{C}_m$, unless $\mathcal{C}_n$ is a parent

```
method buildHierarchy(ClusterList $cl$)
    // $cl = \langle C_i \rangle$ is sorted w.r.t. $\lambda_{C_i}$
    $\lambda_{max} := d$; // $d$ = dimensionality of data space
    for each $\mathcal{C}_i \in cl$ do
        for each $\mathcal{C}_j \in cl$ with $\lambda_{\mathcal{C}_i} < \lambda_{\mathcal{C}_j}$ do
            if $\lambda_{\mathcal{C}_j} = \lambda_{max} \wedge \mathcal{C}_i$.parents=$\emptyset$ then
                $\mathcal{C}_i$.addParent($\mathcal{C}_j$);
            else
                if $\text{CORRDIST}_{\Delta}^{\delta}(\mathcal{C}_i, \mathcal{C}_j) = 0 \wedge$
                  ($\mathcal{C}_i$.parents=$\emptyset \vee$
                  $\neg$ isParent($\mathcal{C}_j$, $\mathcal{C}_i$.parents)) then
                    $\mathcal{C}_i$.addParent($\mathcal{C}_j$);
                end if
            end if
        end for
    end for
```

Figure 3. The method to build the hierarchy of correlation clusters.

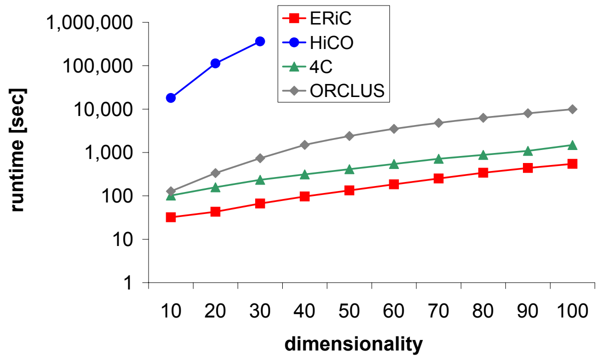
of any cluster $\mathcal{C}_o$ that in turn is already a parent of $\mathcal{C}_m$, because in that case the relationship between $\mathcal{C}_n$ and $\mathcal{C}_m$ is that of a grandparent. The pseudocode of this procedure is depicted in Figure 3.

The resulting hierarchy is visualized using a graph-like representation. An example is depicted in Figure 2(c) showing the hierarchy of correlation clusters in sample data set 2 (cf. Figure 1). In general, the representation is organized top-down w.r.t. the correlation dimensionality similar to a tree but allows multiple inclusions. The "root" of the graph contains all objects in partition $\mathcal{D}_d$. All correlation clusters with equal correlation dimensionality are placed at the same level below the root. Thus, 1D correlation clusters are placed at the bottom level. Each object is placed in that correlation cluster with the smallest correlation dimensionality. An edge between two nodes indicates a (containment) relationship. In fact, a node N represents a cluster of all objects assigned to N as well as all objects assigned to child nodes of N.

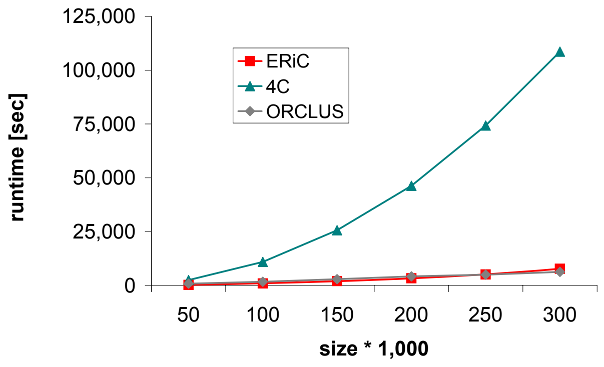
5. Evaluation

5.1. Efficiency

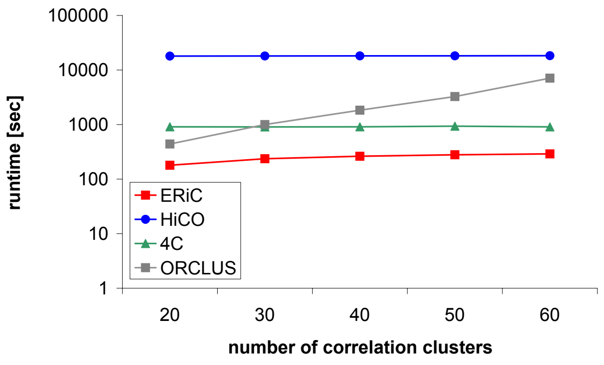
Runtime Complexity. The preprocessing step of ERiC works for each point as follows: First a k-nearest neighbor query is performed, which has a complexity of $O(n)$ since the data set is scanned sequentially. Based on the result of the k-nearest neighbor query, the $d \times d$ covariance matrix is determined. This can be done in $O(k \cdot d^2)$ time. Then



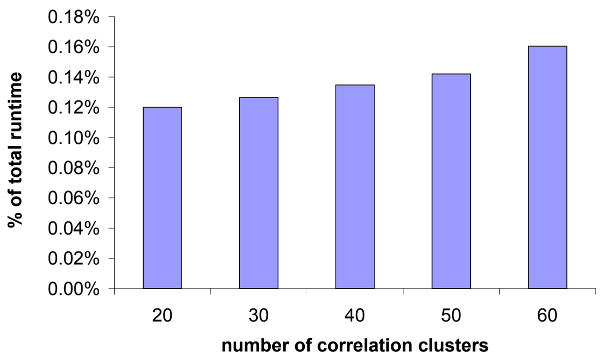
(a) Scalability w.r.t. dimensionality



(b) Scalability w.r.t. size

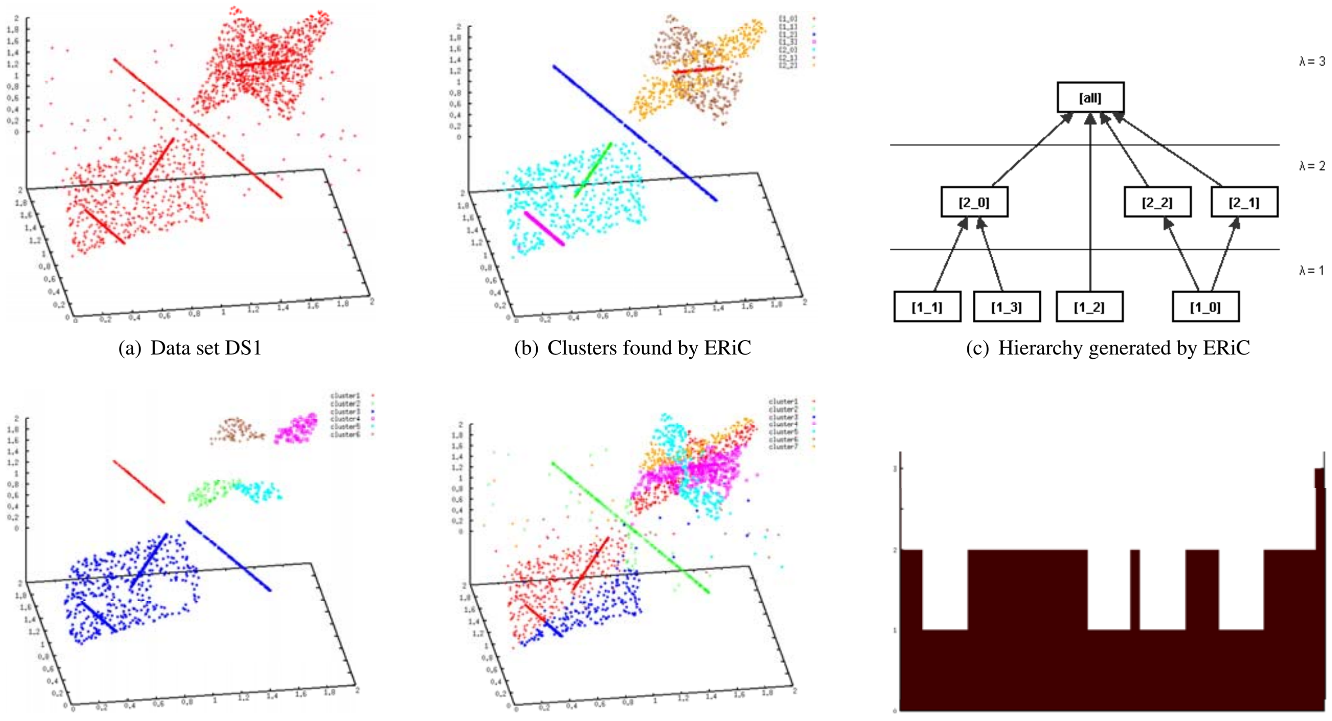


(c) Scalability w.r.t. number of clusters



(d) Runtime of the hierarchy aggregation w.r.t. number of clusters

Figure 4. Scalability results.



(a) Data set DS1 (b) Clusters found by ERiC (c) Hierarchy generated by ERiC

(d) Clusters found by 4C (e) Clusters found by ORCLUS (f) Hierarchy generated by HiCO

Figure 5. Comparative evaluation of different algorithms on 3D synthetic data set DS1.

the covariance matrix is decomposed using PCA which requires $O(d^3)$ time. Thus, for all points together we have a time complexity of $O(n^2 + k \cdot d^2 \cdot n)$ in the first step of ERiC, since $d << k$ as discussed above.

Applying DBSCAN to the data set in the second step of ERiC results in a time complexity of $O(d^3 \cdot n_i^2)$, where n_i is the number of points in partition i. This is due to the fact, that the original DBSCAN has a worst case complexity of $O(n^2)$ on top of the sequential scan. Applying the correlation distance as given in Definition 7, the overall time complexity of DBSCAN is $O(d^3 \cdot n_i^2)$. Assuming on average a uniform distribution of the points over all possible correlation dimensionalities, all partitions contain $\frac{n}{d}$ points. Thus, for d partitions, the required runtime reduces to $O(d^2 \cdot n^2)$.

The hierarchy aggregation considers all pairs of clusters $(\mathcal{C}_i, \mathcal{C}_j)$ associated to different partitions (i.e., $\lambda_i < \lambda_j$) and determines the $\text{CORRDIST}_{\Delta}^{\delta}$ for the corresponding cluster representatives. Let $|\mathcal{C}|$ be the number of clusters. Then the complexity of this method corresponds to $O(|\mathcal{C}|^2 \cdot d^3)$. However, in the experimental evaluation, we show the hierarchy aggregation to require only a marginal runtime compared to the overall runtime of ERiC. This is due to the fact that $|\mathcal{C}| << n$.

Thus, the overall runtime complexity of ERiC can be considered as $O(n^2 \cdot d^2)$.

Experimental Evaluation. The scalability of the competing methods is depicted in Figure 4. Please note the logarithmic scale of the runtime-axis in Figure 4(a). As it can be seen, ERiC scales better than the compared methods w.r.t. the dimensionality. ERiC clearly outperforms 4C and HiCO w.r.t. the size of the data set showing a runtime comparative to ORCLUS. The runtime of HiCO w.r.t. the size of the data set is far above the others and therefore omitted in the chart for clearness. In these both experiments, the objects are uniformly distributed over 9 correlation clusters and noise.

In a third experiment we investigated the impact of the number of clusters on the runtime behaviour using data sets in a 10-dimensional data space containing 10,000 objects uniformly distributed over an increasing number of clusters. As shown in Figure 4(c), the runtime of ERiC, 4C, and HiCO is quite robust w.r.t. the number of correlation clusters, while the runtime of ORCLUS increases considerably. Again, ERiC gains a significant speed-up over its competitors.

Finally, we analyzed the impact of the hierarchy aggregation on the overall runtime of ERiC w.r.t. the number of correlation clusters (see Figure 4(d)). This experiment is based on the data sets of the third experiment.

Overall, the experiments confirm the theoretical considerations presented above.

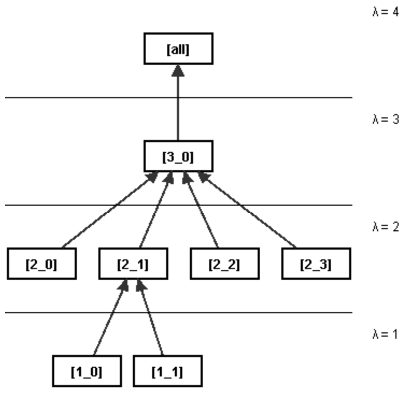
5.2. Accuracy

Synthetic Data. We evaluated the accuracy of ERiC in comparison to ORCLUS, 4C, and HiCO on several synthetic data sets. In all experiments, we optimized the input parameters of all methods in terms of cluster quality and report the best results in order to achieve a fair comparison. Figure 5 illustrates the results of the competitors on a synthetic 3D data set (cf. Figure 5(a)) containing several 1D and 2D correlation clusters with complex hierarchical relationships. The result of ERiC (cf. Figure 5(c)) illustrates the correct and complete hierarchy. One can see at a glance that the data set contains two 1D clusters (lines "1_1" and "1_3") embedded within a 2D cluster (plane "2_0"), one separate 1D cluster (line "1_2"), and a multiple inclusion of one 1D cluster (line "1_0") embedded within two 2D clusters (planes "2_1" and "2_2"). None of the existing state-of-the-art correlation clustering approaches performs equally well. 4C (cf. Figure 5(d)) has problems to separate several clusters ('1_1", "1_3", "2_0", and parts of "1_2") from each other and, on the other hand, splits compact clusters into several parts (e.g. clusters "1_2", "2_1", "2_2", and "1_0"). A similar observation can be made when looking at the results of ORCLUS (cf. Figure 5(e)). Since both 4C and ORCLUS produce a flat clustering, no hierarchy can be derived from their results. Last but not least, the result of HiCO is depicted in Figure 5(f). The obtained reachability plot suggests one big 2D correlation cluster (the valley at level 2) having four 1D correlation clusters embedded (the valleys at level 1). Thus, the three 2D clusters are not visible in the resulting plot. In summary, while ERiC has no problems to reveal the complete hierarchy of correlation clusters and produce all correct clusters, the competitors all fail to produce the true clusters and the proper hierarchy.

We made further experiments on higher dimensional data sets containing correlation clusters with complex hierarchical relationships. In all experiments we made observations similar to those made with data set DS1. While ERiC perfectly discovered the complete hierarchy and produced 100% correct clusters, 4C, ORCLUS, and HiCO had problems to find pure and complete clusters. In addition, HiCO always failed to detect the complete hierarchical relationships.

Real World Data. We evaluated ERiC on the Wages data set[1] consisting of 534 4-dimensional observations from the 1985 Current Population Survey (dimensions include A=age, YE=years of education, YW=years of work experience, and W=wage). The results are shown in Figure 6(a). ERiC found seven clusters. The contents of the clusters are summarized in Figure 6(b). As it can be seen, the derived clusters are rather meaningful.

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



(a) Hierarchy generated by ERiC

cluster	description
1_0	YE = 12, A = 22, YW = 4
1_1	YE = 12, A = 22, YW = 20
2_0	YE = 14, A = YW + 20
2_1	YE = 12, A = YW+18
2_2	YE = 16, A = YW + 22
2_3	YE = 13, A = YW+19
3_0	YE = A - YW - 6

(b) Contents of found clusters

Figure 6. Results of ERiC on the wages data set.

A second data set used for evaluating ERiC is the pendigits data set[2] containing approximately 7,500 16-dimensional points representing certain features of handwritten digits. The objects are labeled according the digit. The resulting hierarchy computed by ERiC is depicted in Figure 7. Interestingly, all clusters found by ERiC are pure, i.e. contain only objects from one class. The clusters forming the observed multiple inclusion also contain objects from the same class.

In summary, our experiments confirmed that ERiC finds meaningful cluster hierarchies allowing for multiple inclusions in real world data sets.

6. Conclusions

Correlation clustering is a data mining task important for many applications. In this paper, we have motivated to search for complex hierarchies of correlation clusters, including the information that lower dimensional correlation clusters are embedded within higher dimensional ones. Since none of the existing algorithms for correlation clustering can reveal the complete hierarchical structure, we pro-

[2]http://www.ics.uci.edu/~mlearn/databases/pendigits/

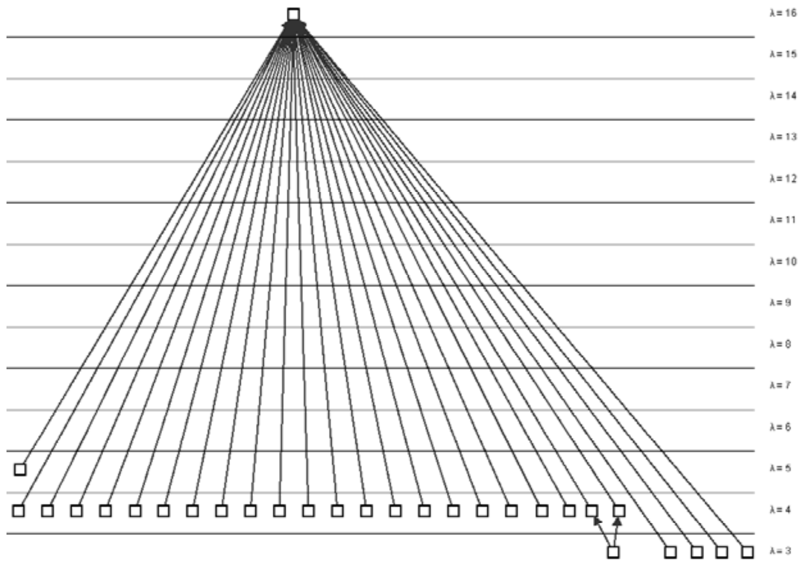


Figure 7. Hierarchy generated by ERiC on pendigits data set.

posed ERiC, a novel clustering algorithm to detect complex hierarchical relationships between correlation clusters also allowing for multiple inclusions. The resulting cluster hierarchy is visualized by means of a clear graph model. We showed theoretically and experimentally that ERiC outperforms existing state-of-the-art correlation clustering algorithms in terms of runtime and accuracy.

References

[1] E. Achtert, C. Böhm, H.-P. Kriegel, P. Kröger, I. Müller-Gorman, and A. Zimek. Finding hierarchies of subspace clusters. In *Proc. PKDD*, 2006.

[2] E. Achtert, C. Böhm, H.-P. Kriegel, P. Kröger, I. Müller-Gorman, and A. Zimek. Detection and visualization of subspace cluster hierarchies. In *Proc. DASFAA*, 2007.

[3] E. Achtert, C. Böhm, H.-P. Kriegel, P. Kröger, and A. Zimek. Robust, complete, and efficient correlation clustering. In *Proc. SDM*, 2007.

[4] E. Achtert, C. Böhm, P. Kröger, and A. Zimek. Mining hierarchies of correlation clusters. In *Proc. SSDBM*, 2006.

[5] 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.

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

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

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

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

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

[11] K. Chakrabarti and S. Mehrotra. Local dimensionality reduction: A new approach to indexing high dimensional spaces. In *Proc. VLDB*, 2000.

[12] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1):1–31, 1977.

[13] 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.

[14] A. Gionis, A. Hinneburg, S. Papadimitriou, and P. Tsaparas. Dimension induced clustering. In *Proc. KDD*, 2005.

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

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

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

[18] 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.

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

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

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

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