in Proc. ACM Int. Conf. on Management of Data (SIGMOD), pp. 455-466, Paris, France, 2004

Computing Clusters of Correlation Connected Objects*

Christian Böhm, Karin Kailing, Peer Kröger, Arthur Zimek

Institute for Computer Science
University of Munich
Oettingenstr. 67, 80538 Munich, Germany
{boehm,kailing,kriegel,kroegerp,zimek}@dbs.informatik.uni-muenchen.de

ABSTRACT

The detection of correlations between different features in a set of feature vectors is a very important data mining task because correlation indicates a dependency between the features or some association of cause and effect between them. This association can be arbitrarily complex, i.e. one or more features might be dependent from a combination of several other features. Well-known methods like the principal components analysis (PCA) can perfectly find correlations which are global, linear, not hidden in a set of noise vectors, and uniform, i.e. the same type of correlation is exhibited in all feature vectors. In many applications such as medical diagnosis, molecular biology, time sequences, or electronic commerce, however, correlations are not global since the dependency between features can be different in different subgroups of the set. In this paper, we propose a method called *4C* (Computing Correlation Connected Clusters) to identify local subgroups of the data objects sharing a uniform but arbitrarily complex correlation. Our algorithm is based on a combination of PCA and density-based clustering (DBSCAN). Our method has a determinate result and is robust against noise. A broad comparative evaluation demonstrates the superior performance of 4C over competing methods such as DBSCAN, CLIQUE and ORCLUS.

1. INTRODUCTION

The tremendous amount of data produced nowadays in various application domains can only be fully exploited by efficient and effective data mining tools. One of the primary data mining tasks is clustering, which aims at partitioning the objects (described by points in a high dimensional feature space) of a data set into dense regions (clusters) separated by regions with low density (noise). Knowing the cluster structure is important and valuable because the different clusters often represent different classes of objects which have previously been unknown. Therefore, the clusters bring additional insight about the stored data set which can be exploited for various purposes such as improved marketing by customer segmentation, determination of homogeneous groups of web users through clustering of web logs, structuring large amounts of text documents by hierarchical clustering, or developing thematic maps from satellite images.

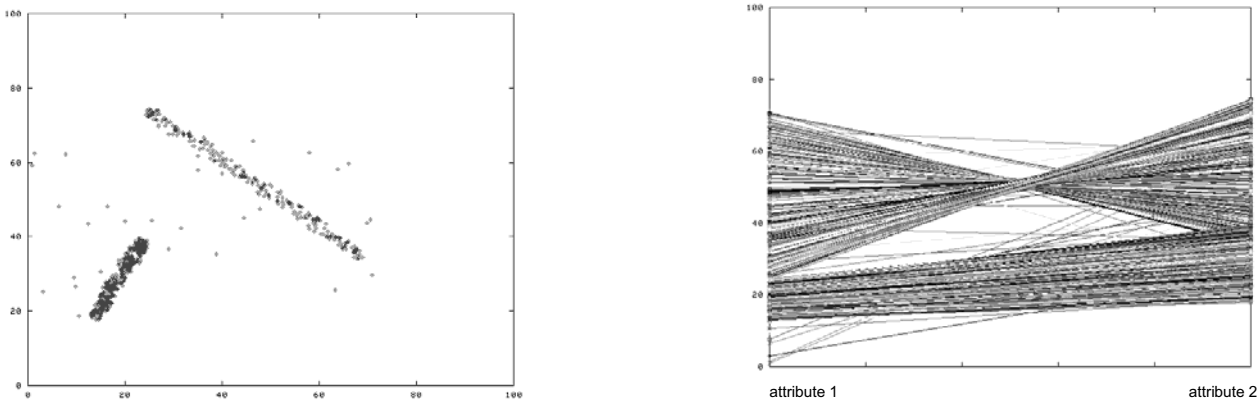
An interesting second kind of hidden information that may be interesting to users are correlations in a data set. A correlation is a linear dependency between two or more features (attributes) of the data set. The most important method for detecting correlations is the principal components analysis (PCA) also known as Karhunen Loèwe transformation. Knowing correlations is also important and valuable because the dimensionality of the data set can be considerably reduced which improves both the performance of similarity search and data mining as well as the accuracy. Moreover, knowing about the existence of a relationship between attributes enables one to detect hidden causalities (e.g. the influence of the age of a patient and the dose rate of medication on the course of his disease or the coregulation of gene expression) or to gain financial advantage (e.g. in stock quota analysis), etc.

Methods such as PCA, however, are restricted, because they can only be applied to the data set as a whole. Therefore, it is only possible to detect correlations which are expressed in all points or almost all points of the data set. For a lot of applications this is not the case. For instance, in the analysis of gene expression, we are facing the problem that a dependency between two genes does only exist under certain conditions. Therefore, the correlation is visible only in a local subset of the data. Other subsets may be either not correlated at all, or they may exhibit completely different kinds of correlation (different features are dependent on each other). The correlation of the whole data set can be weak, even if for local subsets of the data strong correlations exist. Figure 1 shows a simple example, where two subsets of 2-dimensional points exhibit different correlations.

To the best of our knowledge, both concepts of clustering (i.e. finding densely populated subsets of the data) and correlation analysis have not yet been addressed as a combined task for data mining. The most relevant related approach is ORCLUS [1], but since it is k-medoid-based, it is very sen-

*Supported in part by the German Ministry for Education, Science, Research and Technology (BMBF) under grant no. 031U112F within the BFAM (Bioinformatics for the Functional Analysis of Mammalian Genomes) project which is part of the German Genome Analysis Network (NGFN).

SIGMOD 2004, June 13-18, 2004, Paris, France.



(a) 2D view.

(b) Transposed view.

Figure 1: 1-Dimensional Correlation Lines

sitive to noise and the locality of the analyzed correlations is usually too coarse, i.e. the number of objects taken into account for correlation analysis is too large (cf. Section 2 and 6 for a detailed discussion). In this paper, we develop a new method which is capable of detecting local subsets of the data which exhibit strong correlations and which are densely populated (w.r.t. a given density threshold). We call such a subset a *correlation connected cluster*.

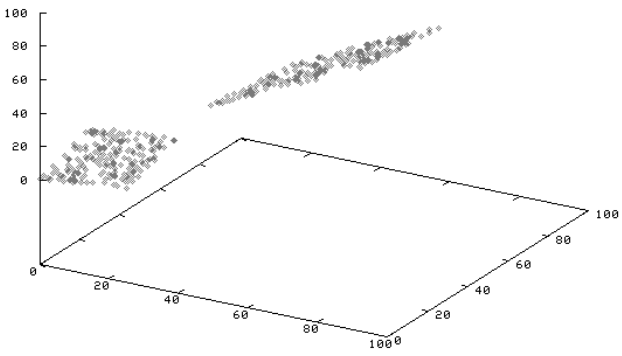
In lots of applications such correlation connected clusters are interesting. For example in E-commerce (recommendation systems or target marketing) where sets of customers with similar behavior need to be detected, one searches for positive linear correlations. In DNA microarray analysis (gene expression analysis) negative linear correlations express the fact that two genes may be coregulated, i.e. if one has a high expression level, the other one is very low and *vice versa*. Usually such a coregulation will only exist in a small subset of conditions or cases, i.e. the correlation will be hidden locally in the data set and cannot be detected by global techniques. Figures 1 and 2 show simple examples how correlation connected clusters can look like. In Figure 1 the attributes exhibit two different forms of linear correlation. We observe that if for some points there is a linear correlation of all attributes, these points are located along a line. Figure 2 presents two examples where an attribute z is correlated to attributes x and y (i.e. $z = a + bx + cy$). In this case the set of points forms a 2-dimensional plane.

As stated above, in this paper, we propose an approach that meets both the goal of clustering and correlation analysis in order to find correlation connected clusters. The remainder is organized as follows: In Section 2 we review and discuss related work. In Section 3 we formalize our notion of correlation connected clusters. Based on this formalization, we present in Section 4 an algorithm called 4C (*C*omputing *C*orrelation *C*onnected *C*lusters) to efficiently compute such correlation connected clusters. In Section 5 we analyze the computational complexity of our algorithm, while Section 6 contains an extensive experimental evaluation of 4C. Section 7 concludes the paper and gives some hints at future work.

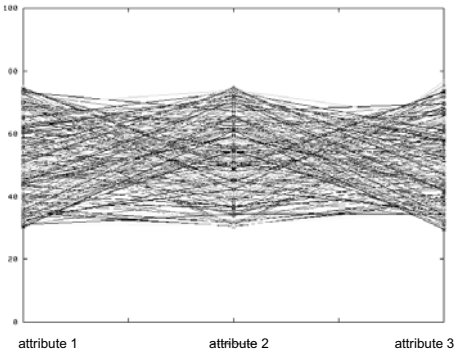
2. RELATED WORK

Traditional clustering algorithms such as k-means or the EM-algorithm search for spatial clusters, which are spherically shaped. In [10] and [5] two algorithms are proposed which are able to find clusters of arbitrary shape. However, these approaches are not able to distinguish between arbitrarily shaped clusters and correlation clusters. In [10] the density-based algorithm DBSCAN is introduced. We will describe the basic concepts of this approach in detail in the next chapter. In [5] the authors propose the algorithm FC (Fractal Clustering) to find clusters of arbitrary shape. The paper presents only experiments for two-dimensional data sets, so it is not clear whether the fractal dimension is really stable in higher dimensions. Furthermore, the shapes of clusters depend on the type of correlation of the involved points. Thus, in the case of linear correlations it is more target-oriented to search for shapes like lines or hyperplanes than for arbitrarily shaped clusters.

As most traditional clustering algorithms fail to detect meaningful clusters in high-dimensional data, in the last years a lot of research has been done in the area of subspace clustering. In the following we are examining to what extent subspace clustering algorithms are able to capture local data correlations and find clusters of correlated objects. The principal axis of correlated data are arbitrarily oriented. In contrast, subspace clustering techniques like CLIQUE [3] and its successors MAFIA [11] or ENCLUS [9] or projected clustering algorithms like PROCLUS [2] and DOC [17] only find axis parallel projections of the data. In the evaluation part we show that CLIQUE as one representative algorithm is in fact not able to find correlation clusters. Therefore we focus our attention to ORCLUS [1] which is a k-medoid related projected clustering algorithm allowing clusters to exist in arbitrarily oriented subspaces. The problem of this approach is that the user has to specify the number of clusters in advance. If this guess does not correspond to the actual number of clusters the results of ORCLUS deteriorate. A second problem, which can also be seen in the evaluation part is noisy data. In this case the clusters found by ORCLUS are far from optimal, since ORCLUS assigns each object to a cluster and thus cannot handle noise efficiently.



(a) 3D view.



(b) Transposed view of one plane.

Figure 2: 2-Dimensional Correlation Planes

Based on the fractal (intrinsic) dimensionality of a data set, the authors of [15] present a global dimensionality reduction method. Correlation in the data leads to the phenomenon that the embedding dimension of a data set (in other words the number of attributes of the data set) and the intrinsic dimension (the dimension of the spatial object represented by the data) can differ a lot. The intrinsic (correlation fractal dimension) is used to reduce the dimensionality of the data. As this approach adopts a global view on the data set and does not account for local data distributions, it cannot capture local subspace correlations. Therefore it is only useful and applicable, if the underlying correlation affects all data objects. Since this is not the case for most real-world data, in [8] a local dimensionality reduction technique for correlated data is proposed which is similar to [1]. The authors focus on identifying correlated clusters for enhancing the indexing of high dimensional data only. Unfortunately, they do not give any hint to what extent their heuristic-based algorithm can also be used to gain new insight into the correlations contained in the data.

In [23] the move-based algorithm FLOC computing near-optimal δ-clusters is presented. A transposed view of the data is used to show the correlations which are captured by the δ-cluster model. Each data object is shown as a curve where the attribute values of each object are connected. Examples can be seen in Figure 1(b) and 2(b). A cluster is regarded as a subset of objects and attributes for which the participating objects show the same (or a similar) tendency rather than being close to each other on the associated subset of dimensions. The δ-cluster model concentrates on two forms of coherence, namely shifting (or addition) and amplification (or production). In the case of amplification coherence for example, the vectors representing the objects must be multiples of each other. The authors state that this can easily be transformed to the problem of finding shifting coherent δ-cluster by applying logarithmic function to each object. Thus they focus on finding shifting coherent δ-clusters and introduce the metric of residue to measure the coherency among objects of a given cluster. An advantage is that thereby they can easily handle missing attribute values. But in contrast to our approach, the δ-cluster model limits itself to a very special form of correlation where all attributes are positively linear correlated. It does not include negative correlations or correlations where one attribute is determined by two or more other attributes. In this cases searching for a trend is no longer possible as can be seen in Figure 1 and 2. Let us note, that such complex dependencies cannot be illustrated by transposed views of the data. The same considerations apply for the very similar p-cluster model introduced in [21] and two extensions presented in [16, 14].

3. THE NOTION OF CORRELATION CONNECTED CLUSTERS

In this section, we formalize the notion of a correlation connected cluster. Let $\mathcal{D}$ be a database of d-dimensional feature vectors ($\mathcal{D} \subseteq \mathbb{R}^d$). An element $P \in \mathcal{D}$ is called point or object. The value of the i-th attribute ($1 \leq i \leq d$) of P is denoted by p_i (i.e. $P = (p_1, \ldots, p_d)^{\mathbf{T}}$). Intuitively, a correlation connected cluster is a dense region of points in the d-dimensional feature space having at least one principal axis with low variation along this axis. Thus, a correlation connected cluster has two different properties: density and correlation. In the following, we will first address these two properties and then merge these ingredients to formalize our notion of correlation connected clusters.

3.1 Density-Connected Sets

The density-based notion is a common approach for clustering used by various clustering algorithms such as DBSCAN [10], DBCLASD [22], DENCLUE [12], and OPTICS [4]. All these methods search for regions of high density in a feature space that are separated by regions of lower density.

A typical density-based clustering algorithm needs two parameters to define the notion of density: First, a parameter μ specifying the minimum number of objects, and second, a parameter ε specifying a volume. These two parameters determine a density threshold for clustering.

Our approach follows the formal definitions of density-based clusters underlying the algorithm DBSCAN. The formal definition of the clustering notion is presented and discussed in full details in [10]. In the following we give a short summary of all necessary definitions.

DEFINITION 1 (ε-NEIGHBORHOOD).
Let $\varepsilon \in \mathbb{R}$ and $O \in \mathcal{D}$. The ε-neighborhood *of O, denoted*

by $\mathcal{N}_\varepsilon(O)$, is defined by

$$\mathcal{N}_\varepsilon(O) = \{X \in \mathcal{D} \mid dist(O, X) \leq \varepsilon\}.$$

Based on the two input parameters ε and μ, dense regions can be defined by means of core objects:

DEFINITION 2 (CORE OBJECT).
Let $\varepsilon \in \mathbb{R}$ and $\mu \in \mathbb{N}$. An object $O \in \mathcal{D}$ is called core object *w.r.t. ε and μ, if its ε-neighborhood contains at least μ objects, formally:*

$$\text{CORE}_{\text{den}}(O) \Leftrightarrow |\mathcal{N}_\varepsilon(O)| \geq \mu.$$

Let us note, that we use the acronym "den" for the density parameters ε and μ. In the following, we omit the parameters ε and μ wherever the context is clear and use "den" instead.

A core object O can be used to expand a cluster, with all the density-connected objects of O. To find these objects the following concepts are used.

DEFINITION 3 (DIRECT DENSITY-REACHABILITY).
Let $\varepsilon \in \mathbb{R}$ and $\mu \in \mathbb{N}$. An object $P \in \mathcal{D}$ is directly density-reachable *from $Q \in \mathcal{D}$ w.r.t. ε and μ, if Q is a core object and P is an element of $\mathcal{N}_\varepsilon(Q)$, formally:*

$$\text{DIRREACH}_{\text{den}}(Q, P) \Leftrightarrow \text{CORE}_{\text{den}}(Q) \wedge P \in \mathcal{N}_\varepsilon(Q).$$

Let us note, that direct density-reachability is symmetric only for core objects.

DEFINITION 4 (DENSITY-REACHABILITY).
Let $\varepsilon \in \mathbb{R}$ and $\mu \in \mathbb{N}$. An object $P \in \mathcal{D}$ is density-reachable *from $Q \in \mathcal{D}$ w.r.t. ε and μ, if there is a chain of objects $P_1, \ldots, Pn \in \mathcal{D}$, $P_1 = Q$, $P_n = P$ such that P_{i+1} is directly density-reachable from P_i, formally:*

$$\begin{aligned}&\text{REACH}_{\text{den}}(Q, P) \Leftrightarrow \\ &\quad \exists P_1, \ldots, Pn \in \mathcal{D} : P_1 = Q \wedge P_n = P \wedge \\ &\quad \forall i \in \{1, \ldots, n-1\} : \text{DIRREACH}_{\text{den}}(P_i, P_{i+1}).\end{aligned}$$

Density-reachability is the transitive closure of direct density-reachability. However, it is still not symmetric in general.

DEFINITION 5 (DENSITY-CONNECTIVITY).
Let $\varepsilon \in \mathbb{R}$ and $\mu \in \mathbb{N}$. An object $P \in \mathcal{D}$ is density-connected *to an object $Q \in \mathcal{D}$ w.r.t. ε and μ, if there is an object $O \in \mathcal{D}$ such that both P and Q are density-reachable from O, formally:*

$$\begin{aligned}&\text{CONNECT}_{\text{den}}(Q, P) \Leftrightarrow \\ &\exists O \in \mathcal{D} : \text{REACH}_{\text{den}}(O, Q) \wedge \text{REACH}_{\text{den}}(O, P).\end{aligned}$$

Density-connectivity is a symmetric relation. A density-connected cluster is defined as a set of density-connected objects which is maximal w.r.t. density-reachability [10].

DEFINITION 6 (DENSITY-CONNECTED SET).
Let $\varepsilon \in \mathbb{R}$ and $\mu \in \mathbb{N}$. A non-empty subset $\mathcal{C} \subseteq \mathcal{D}$ is called a density-connected set *w.r.t. ε and μ, if all objects in $\mathcal{C}$ are density-connected and $\mathcal{C}$ is maximal w.r.t. density-reachability, formally:*

$\text{CONSET}_{\text{den}}(\mathcal{C}) \Leftrightarrow$

(1) Connectivity: $\forall O, Q \in \mathcal{C} : \text{CONNECT}_{\text{den}}(O, Q)$

(2) Maximality: $\forall P, Q \in \mathcal{D} : Q \in \mathcal{C} \wedge \text{REACH}_{\text{den}}(Q, P) \Rightarrow P \in \mathcal{C}$.

Using these concepts DBSCAN is able to detect arbitrarily shaped clusters by one single pass over the data. To do so, DBSCAN uses the fact, that a density-connected cluster can be detected by finding one of its core-objects O and computing all objects which are density reachable from O. The correctness of DBSCAN can be formally proven (cf. lemmata 1 and 2 in [10], proofs in [19]). Although DBSCAN is not in a strong sense deterministic (the run of the algorithm depends on the order in which the points are stored), both the run-time as well as the result (number of detected clusters and association of core objects to clusters) are determinate. The worst case time complexity of DBSCAN is $O(n \log n)$ assuming an efficient index and $O(n^2)$ if no index exists.

3.2 Correlation Sets

In order to identify correlation connected clusters (regions in which the points exhibit correlation) and to distinguish them from usual clusters (regions of high point density only) we are interested in all sets of points with an intrinsic dimensionality that is considerably smaller than the embedding dimensionality of the data space (e.g. a line or a plane in a three or higher dimensional space). There are several methods to measure the intrinsic dimensionality of a point set in a region, such as the fractal dimension or the principal components analysis (PCA). We choose PCA because the fractal dimension appeared to be not stable enough in our first experiments.

The PCA determines the covariance matrix $\mathbf{M} = [m_{ij}]$ with $m_{ij} = \sum_{S \in \mathcal{S}} s_i s_j$ of the considered point set $\mathcal{S}$, and decomposes it into an orthonormal Matrix $\mathbf{V}$ called eigenvector matrix and a diagonal matrix $\mathbf{E}$ called eigenvalue matrix such that $\mathbf{M} = \mathbf{V}\mathbf{E}\mathbf{V}^{\mathbf{T}}$. The eigenvectors represent the principal axes of the data set whereas the eigenvalues represent the variance along these axes. In case of a linear dependency between two or more attributes of the point set (correlation), one or more eigenvalues are close to zero. A set forms a λ-dimensional correlation hyperplane if $d - \lambda$ eigenvalues fall below a given threshold $\delta \approx 0$. Since the eigenvalues of different sets exhibiting different densities may differ a lot in their absolute values, we normalize the eigenvalues by mapping them onto the interval $[0, 1]$. This normalization is denoted by Ω and simply divides each eigenvalue e_i by the maximum eigenvalue e_{max}. We call the eigenvalues e_i with $\Omega(e_i) \leq \delta$ *close to zero.*

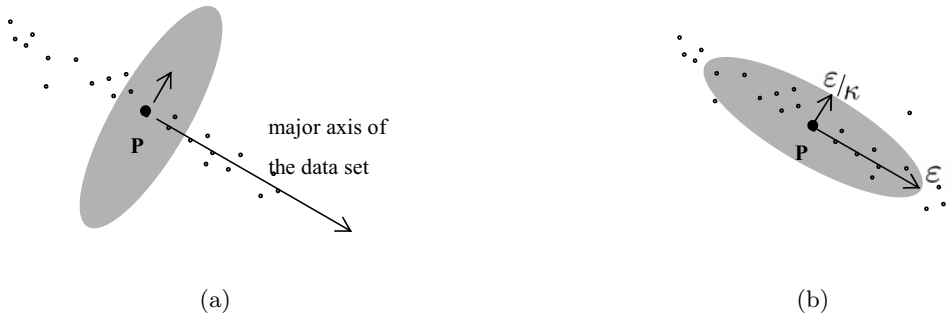
DEFINITION 7 (λ-DIMENSIONAL LINEAR CORRELATION SET).

Let $\mathcal{S} \subseteq \mathcal{D}$, $\lambda \in \mathbb{N}$ ($\lambda \leq d$), $EV = e_1, ..., e_d$ the eigenvalues of $\mathcal{S}$ in descending order (i.e. $e_i \geq e_{i+1}$) and $\delta \in \mathbb{R}$ ($\delta \approx 0$). $\mathcal{S}$ forms an λ-dimensional linear correlation set *w.r.t. δ if at least $d - \lambda$ eigenvalues of $\mathcal{S}$ are close to zero, formally:*

$$\text{CORSET}^{\lambda}_{\delta}(\mathcal{S}) \Leftrightarrow |\{e_i \in EV \mid \Omega(e_i) \leq \delta\}| \geq d - \lambda.$$

where $\Omega(e_i) = e_i / e_1$.

This condition states that the variance of $\mathcal{S}$ along $d - \lambda$ principal axes is low and therefore the objects of $\mathcal{S}$ form an λ-dimensional hyperplane. We drop the index λ and speak of a correlation set in the following wherever it is clear from context.



(a) (b)

Figure 3: Correlation ε-neighborhood of a point P according to (a) $\mathbf{M}_P$ and (b) $\hat{\mathbf{M}}_P$.

DEFINITION 8 (CORRELATION DIMENSION). *Let $\mathcal{S} \in \mathcal{D}$ be a linear correlation set w.r.t. $\delta \in \mathbb{N}$. The number of eigenvalues with $e_i > \delta$ is called* correlation dimension, *denoted by* $\text{CORDIM}(\mathcal{S})$.

Let us note, that if $\mathcal{S}$ is a λ-dimensional linear correlation set, then $\text{CORDIM}(\mathcal{S}) \leq \lambda$. The correlation dimension of a linear correlation set $\mathcal{S}$ corresponds to the intrinsic dimension of $\mathcal{S}$.

3.3 Clusters as Correlation-Connected Sets

A correlation connected cluster can be regarded as a maximal set of density-connected points that exhibit uniform correlation. We can formalize the concept of correlation connected sets by merging the concepts described in the previous two subsections: density-connected sets (cf. Definition 6) and correlation sets (cf. Definition 7). The intuition of our formalization is to consider those points as core objects of a cluster which have an appropriate correlation dimension in their neighborhood. Therefore, we associate each point P with a similarity matrix $\mathbf{M}_P$ which is determined by PCA of the points in the ε-neighborhood of P. For convenience we call $\mathbf{V}_P$ and $\mathbf{E}_P$ the eigenvectors and eigenvalues of P, respectively. A point P is inserted into a cluster if it has the same or a similar similarity matrix like the points in the cluster. To achieve this goal, our algorithm looks for points that are close to the principal axis (or axes) of those points which are already in the cluster. We will define a similarity measure $\hat{\mathbf{M}}_P$ for the efficient search of such points.

We start with the formal definition of the covariance matrix $\mathbf{M}_P$ associated with a point P.

DEFINITION 9 (COVARIANCE MATRIX). *Let $P \in \mathcal{D}$. The matrix $\mathbf{M}_P = [m_{ij}]$ with*

$$m_{ij} = \sum_{S \in \mathcal{N}_\varepsilon(P)} s_i s_j \qquad (1 \leq i, j \leq d)$$

is called the covariance matrix *of the point P. $\mathbf{V}_P$ and $\mathbf{E}_P$ (with $\mathbf{M}_P = \mathbf{V}_P \mathbf{E}_P \mathbf{V}_P^{\mathbf{T}}$) as determined by PCA of $\mathbf{M}_P$ are called the eigenvectors and eigenvalues of the point P, respectively.*

We can now define the new similarity measure $\hat{\mathbf{M}}_P$ which searches points in the direction of highest variance of $\mathbf{M}_P$ (the major axes). Theoretically, $\mathbf{M}_P$ could be directly used as a similarity measure, i.e.

$$dist_{\mathbf{M}_P}(P,Q) = \sqrt{(P-Q)\mathbf{M}_P(P-Q)^{\mathbf{T}}} \quad \text{where } P, Q \in \mathcal{D}.$$

Figure 3(a) shows the set of points which lies in an ε-neighborhood of the point using M_P as similarity measure. The distance measure puts high weights on those axes with a high variance whereas directions with a low variance are associated with low weights. This is usually desired in similarity search applications where directions of high variance have a high distinguishing power and, in contrast, directions of low variance are negligible.

Obviously, for our purpose of detecting correlation clusters, we need quite the opposite. We want so search for points in the direction of highest variance of the data set. Therefore, we need to assign low weights to the direction of highest variance in order to shape the ellipsoid such that it reflects the data distribution (cf. Figure 3(b)). The solution is to change large eigenvalues into smaller ones and *vice versa*. We use two fixed values, 1 and a parameter $\kappa \gg 1$ rather than e.g. inverting the eigenvalues in order to avoid problems with singular covariance matrices. The number 1 is a natural choice because the corresponding semi-axes of the ellipsoid are then epsilon. The parameter κ controls the "thickness" of the λ-dimensional correlation line or plane, i.e. the tolerated deviation.

This is formally captured in the following definition:

DEFINITION 10 (CORRELATION SIMILARITY MATRIX). *Let $P \in \mathcal{D}$ and $\mathbf{V}_P$, $\mathbf{E}_P$ the corresponding eigenvectors and eigenvalues of the point P. Let $\kappa \in \mathbb{R}$ be a constant with $\kappa \gg 1$. The new eigenvalue matrix $\hat{\mathbf{E}}_P$ with entries $\hat{e}_i$ $(i = 1, \ldots d)$ is computed from the eigenvalues $e_1, \ldots, e_d$ in $\mathbf{E}_P$ according to the following rule:*

$$\hat{e}_i = \begin{cases} 1 & if \quad \Omega(e_i) > \delta \\ \kappa & if \quad \Omega(e_i) \leq \delta \end{cases}$$

where Ω is the normalization of the eigenvalues onto $[0,1]$ as described above. The matrix $\hat{\mathbf{M}}_P = \mathbf{V}_P \hat{\mathbf{E}}_P \mathbf{V}_P^{\mathbf{T}}$ is called the correlation similarity matrix. *The* correlation similarity measure *associated with point P is denoted by*

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

Figure 3(b) shows the ε-neighborhood according to the correlation similarity matrix $\hat{\mathbf{M}}_P$. As described above, the parameter κ specifies how much deviation from the correlation is allowed. The greater the parameter κ, the tighter and clearer the correlations which will be computed. It empirically turned out that our algorithm presented in Section

(a) (b)

Figure 4: Symmetry of the correlation ε-neighborhood: (a) $P \in \mathcal{N}_\varepsilon^{\hat{\mathbf{M}}_Q}(Q)$. (b) $P \notin \mathcal{N}_\varepsilon^{\hat{\mathbf{M}}_Q}(Q)$.

4 is rather insensitive to the choice of κ. A good suggestion is to set $\kappa = 50$ in order to achieve satisfying results, thus — for the sake of simplicity — we omit the parameter κ in the following.

Using this similarity measure, we can define the notions of correlation core objects and correlation reachability. However, in order to define correlation-connectivity as a symmetric relation, we face the problem that the similarity measure in Definition 10 is not symmetric, because $dist_P(P,Q) = dist_Q(Q,P)$ does in general not hold (cf. Figure 4(b)). Symmetry, however, is important to avoid ambiguity of the clustering result. If an asymmetric similarity measure is used in DBSCAN a different clustering result can be obtained depending on the order of processing (e.g. which point is selected as the starting object) because the symmetry of density-connectivity depends on the symmetry of direct density-reachability for core-objects. Although the result is typically not seriously affected by this ambiguity effect we avoid this problem easily by an extension of our similarity measure which makes it symmetric. The trick is to consider both similarity measures, $dist_P(P,Q)$ as well as $dist_Q(P,Q)$ and to combine them by a suitable arithmetic operation such as the maximum of the two. Based on these considerations, we define the correlation ε-neighborhood as a symmetric concept:

Definition 11 (correlation ε-neighborhood). *Let $\varepsilon \in \mathbb{R}$. The* correlation ε-neighborhood *of an object $O \in \mathcal{D}$, denoted by $\mathcal{N}_\varepsilon^{\hat{\mathbf{M}}_O}(O)$, is defined by:*

$$\mathcal{N}_\varepsilon^{\hat{\mathbf{M}}_O}(O) = \{X \in \mathcal{D} \mid \max\{dist_O(O,X), dist_X(X,O)\} \leq \varepsilon\}.$$

The symmetry of the correlation ε-neighborhood is illustrated in Figure 4. Correlation core objects can now be defined as follows.

Definition 12 (correlation core object). *Let $\varepsilon, \delta \in \mathbb{R}$ and $\mu, \lambda \in \mathbb{N}$. A point $O \in D$ is called* correlation core object *w.r.t. ε, μ, δ, and λ (denoted by $\text{CORE}_{\text{den}}^{\text{cor}}(O)$), if its ε-neighborhood is a λ-dimensional linear correlation set and its correlation ε-neighborhood contains at least μ points, formally:*

$$\text{CORE}_{\text{den}}^{\text{cor}}(O) \Leftrightarrow \text{CORSET}_\delta^\lambda(\mathcal{N}_\varepsilon(P)) \wedge |\mathcal{N}_\varepsilon^{\hat{\mathbf{M}}_P}(P)| \geq \mu.$$

Let us note that in $\text{CORE}_{\text{den}}^{\text{cor}}$ the acronym "cor" refers to the correlation parameters δ and λ. In the following, we omit the parameters ε, μ, δ, and λ wherever the context is clear and use "den" and "cor" instead.

Definition 13 (Direct correlation-reachability). *Let $\varepsilon, \delta \in \mathbb{R}$ and $\mu, \lambda \in \mathbb{N}$. A point $P \in \mathcal{D}$ is* direct correlation-reachable *from a point $Q \in \mathcal{D}$ w.r.t. ε, μ, δ, and λ (denoted by $\text{DIRREACH}_{\text{den}}^{\text{cor}}(Q,P)$) if Q is a correlation core object, the correlation dimension of $\mathcal{N}_\varepsilon(P)$ is at least λ, and $P \in \mathcal{N}_\varepsilon^{\hat{\mathbf{M}}_Q}(Q)$, formally:*

$\text{DIRREACH}_{\text{den}}^{\text{cor}}(Q,P) \Leftrightarrow$

(1) $\text{CORE}_{\text{den}}^{\text{cor}}(Q)$

(2) $\text{CORDIM}(\mathcal{N}_\varepsilon(P)) \leq \lambda$

(3) $P \in \mathcal{N}_\varepsilon^{\hat{\mathbf{M}}_Q}(Q)$.

Correlation-reachability is symmetric for correlation core objects. Both objects P and Q must find the other object in their corresponding correlation ε-neighborhood.

Definition 14 (correlation-reachability). *Let $\varepsilon, \delta \in \mathbb{R}$ ($\delta \approx 0$) and $\mu, \lambda \in \mathbb{N}$. An object $P \in \mathcal{D}$ is* correlation-reachable *from an object $Q \in \mathcal{D}$ w.r.t. ε, μ, δ, and λ (denoted by $\text{REACH}_{\text{den}}^{\text{cor}}(Q,P)$), if there is a chain of objects $P_1, \cdots P_n$ such that $P_1 = Q, P_n = P$ and P_{i+1} is direct correlation-reachable from P_i, formally:*

$$\begin{aligned}&\text{REACH}_{\text{den}}^{\text{cor}}(Q,P) \Leftrightarrow \\ &\quad \exists P_1, \ldots, P_n \in \mathcal{D} : P_1 = Q \wedge P_n = P \wedge \\ &\quad \forall i \in \{1, \ldots, n-1\} : \text{DIRREACH}_{\text{den}}^{\text{cor}}(P_i, P_{i+1}).\end{aligned}$$

It is easy to see, that correlation-reachability is the transitive closure of direct correlation-reachability.

Definition 15 (correlation-connectivity). *Let $\varepsilon \in \mathbb{R}$ and $\mu \in \mathbb{N}$. An object $P \in \mathcal{D}$ is* correlation-connected *to an object $Q \in \mathcal{D}$ if there is an object $O \in \mathcal{D}$ such that both P and Q are correlation-reachable from O, formally:*

$$\begin{aligned}&\text{CONNECT}_{\text{den}}^{\text{corr}}(Q,P) \Leftrightarrow \\ &\exists o \in \mathcal{D} : \text{REACH}_{\text{den}}^{\text{corr}}(O,Q) \wedge \text{REACH}_{\text{den}}^{\text{corr}}(O,P).\end{aligned}$$

Correlation-connectivity is a symmetric relation. A correlation-connected cluster can now be defined as a maximal correlation-connected set:

DEFINITION 16 (CORRELATION-CONNECTED SET).
Let $\varepsilon, \delta \in \mathbb{R}$ and $\mu, \lambda \in \mathbb{N}$. A non-empty subset $\mathcal{C} \subseteq \mathcal{D}$ is called a density-connected set *w.r.t. ε, μ, δ, and λ, if all objects in $\mathcal{C}$ are density-connected and $\mathcal{C}$ is maximal w.r.t. density-reachability, formally:*

$\text{CONSET}^{\text{cor}}_{\text{den}}(\mathcal{C}) \Leftrightarrow$

(1) Connectivity: $\forall O, Q \in \mathcal{C} : \text{CONNECT}^{\text{cor}}_{\text{den}}(O, Q)$

(2) Maximality: $\forall P, Q \in \mathcal{D} : Q \in \mathcal{C} \wedge \text{REACH}^{\text{cor}}_{\text{den}}(Q, P) \Rightarrow P \in \mathcal{C}$.

The following two lemmata are important for validating the correctness of our clustering algorithm. Intuitively, they state that we can discover a correlation-connected set for a given parameter setting in a two-step approach: First, choose an arbitrary correlation core object O from the database. Second, retrieve all objects that are correlation-reachable from O. This approach yields the density-connected set containing O.

LEMMA 1.
Let $P \in \mathcal{D}$. If P is a correlation core object, then the set of objects, which are correlation-reachable from P is a correlation-connected set, formally:

$$\text{CORE}^{\text{cor}}_{\text{den}}(P) \wedge \mathcal{C} = \{O \in \mathcal{D} \mid \text{REACH}^{\text{cor}}_{\text{den}}(P, O)\} \Rightarrow \text{CONSET}^{\text{cor}}_{\text{den}}(\mathcal{C}).$$

PROOF.
(1) $\mathcal{C} \neq \emptyset$:
By assumption, $\text{CORE}^{\text{cor}}_{\text{den}}$(P) and thus, $\text{CORDIM}(\mathcal{N}_\varepsilon(P)) \leq \lambda$.
$\Rightarrow \text{DIRREACH}^{\text{cor}}_{\text{den}}(P, P)$
$\Rightarrow \text{REACH}^{\text{cor}}_{\text{den}}(P, P)$
$\Rightarrow P \in \mathcal{C}$.

(2) Maximality:
Let $X \in \mathcal{C}$ and $Y \in \mathcal{D}$ and $\text{REACH}^{\text{cor}}_{\text{den}}(X, Y)$.
$\Rightarrow \text{REACH}^{\text{cor}}_{\text{den}}(P, X) \wedge \text{REACH}^{\text{cor}}_{\text{den}}(X, Y)$
$\Rightarrow \text{REACH}^{\text{cor}}_{\text{den}}(P, Y)$ (since correlation reachability is a transitive relation).
$\Rightarrow Y \in \mathcal{C}$.

(3) Connectivity:
$\forall X, Y \in \mathcal{C} : \text{REACH}^{\text{cor}}_{\text{den}}(P, X) \wedge \text{REACH}^{\text{cor}}_{\text{den}}(P, Y)$
$\Rightarrow \text{CONNECT}^{\text{cor}}_{\text{den}}(X, Y)$ (via P). □

LEMMA 2.
Let $\mathcal{C} \subseteq \mathcal{D}$ be a correlation-connected set. Let $P \in \mathcal{C}$ be a correlation core object. Then $\mathcal{C}$ equals the set of objects which are correlation-reachable from P, formally:

$$\text{CONSET}^{\text{cor}}_{\text{den}}(\mathcal{C}) \wedge P \in \mathcal{C} \wedge \text{CORE}^{\text{cor}}_{\text{den}}(P) \Rightarrow \mathcal{C} = \{O \in \mathcal{D} \mid \text{REACH}^{\text{cor}}_{\text{den}}(P, O)\}.$$

PROOF.
Let $\bar{\mathcal{C}} = \{O \in \mathcal{D} \mid \text{REACH}^{\text{cor}}_{\text{den}}(P, O)\}$. We have to show that $\bar{\mathcal{C}} = \mathcal{C}$:

(1) $\bar{\mathcal{C}} \subseteq \mathcal{C}$: obvious from definition of $\bar{\mathcal{C}}$.

(2) $\mathcal{C} \subseteq \bar{\mathcal{C}}$: Let $Q \in \mathcal{C}$. By assumption, $P \in \mathcal{C}$ and $\text{CONSET}^{\text{cor}}_{\text{den}}(\mathcal{C})$.
$\Rightarrow \exists O \in \mathcal{C} : \text{REACH}^{\text{cor}}_{\text{den}}(O, P) \wedge \text{REACH}^{\text{cor}}_{\text{den}}(O, Q)$
$\Rightarrow \text{REACH}^{\text{cor}}_{\text{den}}(P, O)$ (since both O and P are correlation core objects and correlation reachability is symmetric for correlation core objects.
$\Rightarrow \text{REACH}^{\text{cor}}_{\text{den}}(P, Q)$ (transitivity of correlation-reachability)
$\Rightarrow Q \in \bar{\mathcal{C}}$. □

algorithm 4C($\mathcal{D}$, ε, μ, λ, δ)

// assumption: each object in $\mathcal{D}$ is marked as unclassified

for each unclassified $O \in \mathcal{D}$ **do**

STEP 1. test $\text{CORE}^{\text{cor}}_{\text{den}}(O)$ **predicate:**

compute $\mathcal{N}_\varepsilon(O)$;
if $|\mathcal{N}_\varepsilon(O)| \geq \mu$ **then**
compute $\mathbf{M}_O$;
if $\text{CORDIM}(\mathcal{N}_\varepsilon(O)) \leq \lambda$ **then**
compute $\hat{\mathbf{M}}_O$ and $\mathcal{N}_\varepsilon^{\hat{\mathbf{M}}_O}(O)$;
test $|\mathcal{N}_\varepsilon^{\hat{\mathbf{M}}_O}(O)| \geq \mu$;

STEP 2.1. if $\text{CORE}^{\text{cor}}_{\text{den}}(O)$ **expand a new cluster:**

generate new clusterID;
insert all $X \in \mathcal{N}_\varepsilon^{\hat{\mathbf{M}}_O}(O)$ into queue Φ;
while $\Phi \neq \emptyset$ **do**
Q = first object in Φ;
compute $\mathcal{R} = \{X \in \mathcal{D} \mid \text{DIRREACH}^{\text{cor}}_{\text{den}}(Q, X)\}$;
for each $X \in \mathcal{R}$ **do**
if X is unclassified or noise **then**
assign current clusterID to X
if X is unclassified **then**
insert X into Φ;
remove Q from Φ;

STEP 2.2. if not $\text{CORE}^{\text{cor}}_{\text{den}}(O)$ O **is noise:**

mark O as noise;

end.

Figure 5: Pseudo code of the 4C algorithm.

4. ALGORITHM 4C

In the following we describe the algorithm 4C, which performs one pass over the database to find all correlation clusters for a given parameter setting. The pseudocode of the algorithm 4C is given in Figure 5. At the beginning each object is marked as unclassified. During the run of 4C all objects are either assigned a certain cluster identifier or marked as noise. For each object which is not yet classified, 4C checks whether this object is a correlation core object (see STEP 1 in Figure 5). If the object is a correlation core object the algorithm expands the cluster belonging to this object (STEP 2.1). Otherwise the object is marked as noise (STEP 2.2). To find a new cluster 4C starts in STEP 2.1 with an arbitrary correlation core object O and searches for all objects that are correlation-reachable from O. This is sufficient to find the whole cluster containing the object O, due to Lemma 2. When 4C enters STEP 2.1 a new cluster identifier "clusterID" is generated which will be assigned to all objects found in STEP 2.1. 4C begins by inserting all objects in the correlation ε-neighborhood of object O into a queue. For each object in the queue it computes all directly correlation reachable objects and inserts those objects into the queue which are still unclassified. This is repeated until the queue is empty.

As discussed in Section 3 the results of 4C do not depend on the order of processing, i.e. the resulting clustering (number of clusters and association of core objects to clusters) is determinate.

5. COMPLEXITY ANALYSIS

The computational complexity with respect to the number of data points as well as the dimensionality of the data space is an important issue because the proposed algorithms are typically applied to large data sets of high dimensionality. The idea of our correlation connected clustering method is founded on DBSCAN, a density based clustering algorithm for Euclidean data spaces. The complexity of the original DBSCAN algorithm depends on the existence of an index structure for high dimensional data spaces. The worst case complexity is $O(n^2)$, but the existence of an efficient index reduces the complexity to $O(n \log n)$ [10]. DBSCAN is linear in the dimensionality of the data set for the Euclidean distance metric. If a quadratic form distance metric is applied instead of Euclidean (which enables user adaptability of the distance function), the time complexity of DBSCAN is $O(d^2 \cdot n \log n)$. In contrast, subspace clustering methods such as CLIQUE are known to be exponential in the number of dimensions [1, 3].

We begin our analysis with the assumption of no index structure.

LEMMA 3. *The overall worst-case time complexity of our algorithm on top of the sequential scan of the data set is* $O(d^2 \cdot n^2 + d^3 \cdot n)$.

PROOF. Our algorithm has to associate each point of the data set with a similarity measure that is used for searching neighbors (cf. Definition 10). We assume that the corresponding similarity matrix must be computed once for each point, and it can be held in the cache until it is no more needed (it can be easily decided whether or not the similarity matrix can be safely discarded). The covariance matrix is filled with the result of a Euclidean range query which can be evaluated in $O(d \cdot n)$ time. Then the matrix is decomposed using PCA which requires $O(d^3)$ time. For all points together, we have $O(d \cdot n^2 + d^3 \cdot n)$.
Checking the correlation core point property according to Definition 12, and expanding a correlation connected cluster requires for each point the evaluation of a range query with a quadratic form distance measure which can be done in $O(d^2 \cdot n)$. For all points together (including the above cost for the determination of the similarity matrix), we obtain an worst-case time complexity of $O(d^2 \cdot n^2 + d^3 \cdot n)$. □

Under the assumption that an efficient index structure for high dimensional data spaces [7, 6] is available, the complexity of all range queries is reduced from $O(n)$ to $O(\log n)$. Let us note that we can use Euclidean range queries as a filter step for the quadratic form range queries because no semi-axis of the corresponding ellipsoid exceeds ε. Therefore, the overall time complexity in this case is given as follows:

LEMMA 4. *The overall worst case time complexity of our algorithm on top of an efficient index structure for high dimensional data is* $O(d^2 \cdot n \log n + d^3 \cdot n)$.

PROOF. Analogous to Lemma 3. □

6. EVALUATION

In this section, we present a broad evaluation of 4C. We implemented 4C as well as the three comparative methods DBSCAN, CLIQUE, and ORCLUS in JAVA. All experiments were run on a Linux workstation with a 2.0 GHz CPU and 2.0 GB RAM.

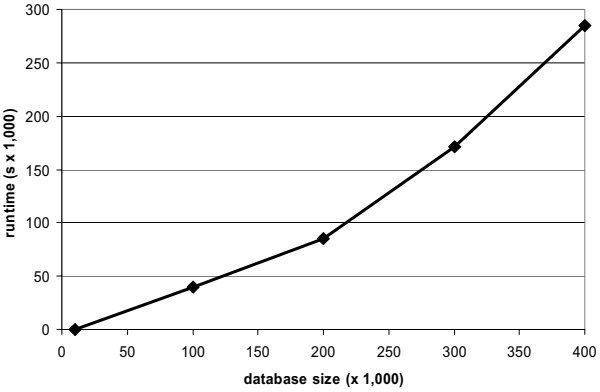


Figure 6: Scalability against database size.

6.1 Efficiency

According to Section 5 the runtime of 4C scales superlinear with the number of input records. This is illustrated in Figure 6 showing the results of 4C applied to synthetic 2-dimensional data of variable size.

6.2 Effectiveness

We evaluated the effectiveness of 4C on several synthetic data sets as well as on real world data sets including gene expression data and metabolome data. In addition, we compared the quality of the results of our method to the quality of the results of DBSCAN, ORCLUS, and CLIQUE. In all our experiments, we set the parameter $\kappa = 50$ as suggested in Section 3.3.

6.2.1 Synthetic Data Sets

We first applied 4C on several synthetic data sets (with $2 \leq d \leq 30$) consisting of several dense, linear correlations. In all cases, 4C had no problems to identify the correlation-connected clusters. As an example, Figure 7 illustrates the transposed view of the three clusters and the noise 4C found on a sample 10-dimensional synthetic data set consisting of approximately 1,000 points.

6.2.2 Real World Data Sets

Gene Expression Data. We applied 4C to the gene expression data set of [20]. The data set is derived from time series experiments on the yeast mitotic cell cycle. The expression levels of approximately 3000 genes are measured at 17 different time slots. Thus, we face a 17-dimensional data space to search for correlations indicating coregulated genes. 4C found 60 correlation connected clusters with few coregulated genes (10-20). Such small cluster sizes are quite reasonable from a biological perspective. The transposed views of four sample clusters are depicted in Figure 8. All four clusters exhibit simple linear correlations on a subset of their attributes. Let us note, that we also found other linear correlations which are rather complex to visualize. We also analyzed the results of our correlation clusters (based on the publicly available information resources on the yeast genome [18]) and found several biologically important implications. For example one cluster consists of several genes coding for proteins related to the assembly of the spindle pole required for mitosis (e.g. KIP1, SLI15, SPC110, SPC25, and NUD1). Another cluster contains sev-

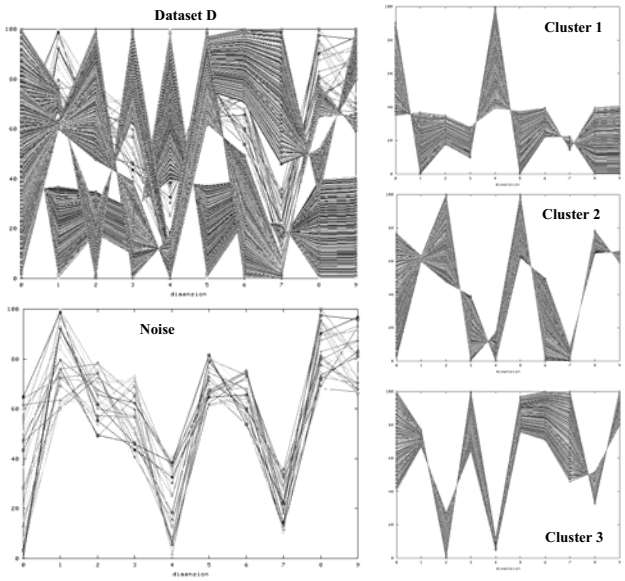


Figure 7: Clusters found by 4C on 10D synthetic data set. Parameters: $\varepsilon = 10.0$, $\mu = 5$, $\lambda = 2$, $\delta = 0.1$.

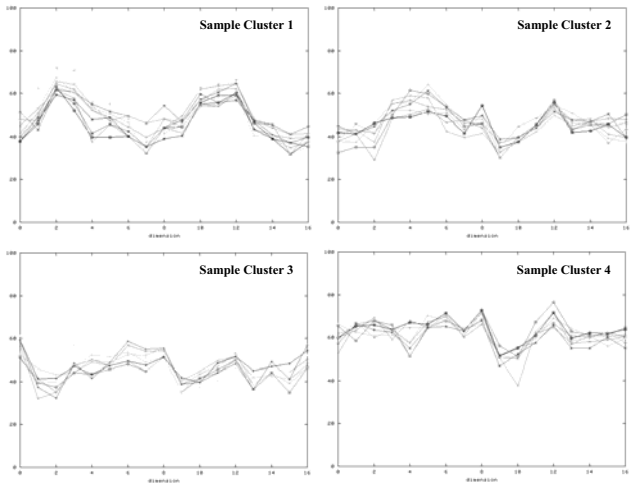


Figure 8: Sample clusters found by 4C on the gene expression data set. Parameters: $\varepsilon = 25.0$, $\mu = 8$, $\lambda = 8$, $\delta = 0.01$.

eral genes coding for structural constituents of the ribosome (e.g. RPL4B, RPL15A, RPL17B, and RPL39). The functional relationships of the genes in the clusters confirm the significance of the computed coregulation.

Metabolome Data. We applied 4C on a metabolome data set [13]. The data set consists of the concentrations of 43 metabolites in 2,000 human newborns. The newborns were labeled according to some specific metabolic diseases. Thus, the data set consists of 2,000 data points with $d = 43$. 4C detected six correlation connected sets which are visualized in Figure 9. Cluster one and two (in the lower left corner marked with "control") consists of healthy newborns whereas the other clusters consists of newborns having one specific disease (e.g. "PKU" or "LCHAD"). The group of newborns suffering from "PKU" was split in three clusters. Several ill as well as healthy newborns were classified as noise.

6.2.3 Comparisons to Other Methods

We compared the effectiveness of 4C with related clustering methods, in particular the density-based clustering algorithm DBSCAN, the subspace clustering algorithm CLIQUE, and the projected clustering algorithm ORCLUS. For that purpose, we applied these methods on several synthetic datasets including 2-dimensional datasets and higher dimensional datasets ($d = 10$).

Comparison with DBSCAN. The clusters found by DBSCAN and 4C applied on the 2-dimensional data sets are depicted in Figure 10. In both cases, DBSCAN finds clusters which do not exhibit correlations (and thus are not detected by 4C). In addition, DBSCAN cannot distinguish varying correlations which overlap (e.g. both correlations in data set B in Figure 10) and treat such clusters as one density-connected set, whereas 4C can differentiate such correlations. We gain similar observations when we applied DBSCAN and 4C on the higher dimensional data sets. Let us note, that these results are not astonishing since DBSCAN only searches for density connected sets but does not search for correlations and thus cannot be applied to the task of finding correlation connected sets.

Comparison with CLIQUE. A comparison of 4C with CLIQUE gained similar results. CLIQUE finds clusters in subspaces which do not exhibit correlations (and thus are not detected by 4C). On the other hand, CLIQUE is usually limited to axis-parallel clusters and thus cannot detect arbitrary correlations. These observations occur especially with higher dimensional data ($d \geq 10$ in our tests). Again, these results are not astonishing since CLIQUE only searches for axis-parallel subspace clusters (dense projections) but does not search for correlations. This empirically supported the suspicion that CLIQUE cannot be applied to the task of finding correlation connected sets.

Comparison with ORCLUS. A comparison of 4C with ORCLUS resulted in quite different observations. In fact, ORCLUS computes clusters of correlated objects. However, since it is a k-medoid based, it suffers from the following two drawbacks: First, the choice of k is a rather hard task for real-world data sets. Even for synthetic data sets, where we knew the number of clusters beforehand, ORCLUS often performs better with a slightly different value of k. Second, ORCLUS is rather sensitive to noise which often appears in real-world data sets. Since all objects have to be assigned to a cluster, the locality of the analyzed correlations is often too coarse (i.e. the subsets of the points taken into account for correlation analysis are too large). As a consequence, the correlation clusters are often blurred by noise objects and thus are hard to obtain from the resulting output. Figure 11 illustrates a sample 3-dimensional synthetic data set, the clusters found by 4C are marked by black lines. Figure 12 depicts the objects in each cluster found by ORCLUS ($k = 3$ yields the best result) separately. It can be seen, that the correlation clusters are — if detected — blurred by noise objects. When we applied ORCLUS on higher dimensional data sets ($d = 10$) the choice of k became even more complex and the problem of noise objects blurring the clusters (i.e. too coarse locality) simply cumulated in the fact that ORCLUS often could not detect correlation clusters in high-dimensional data.

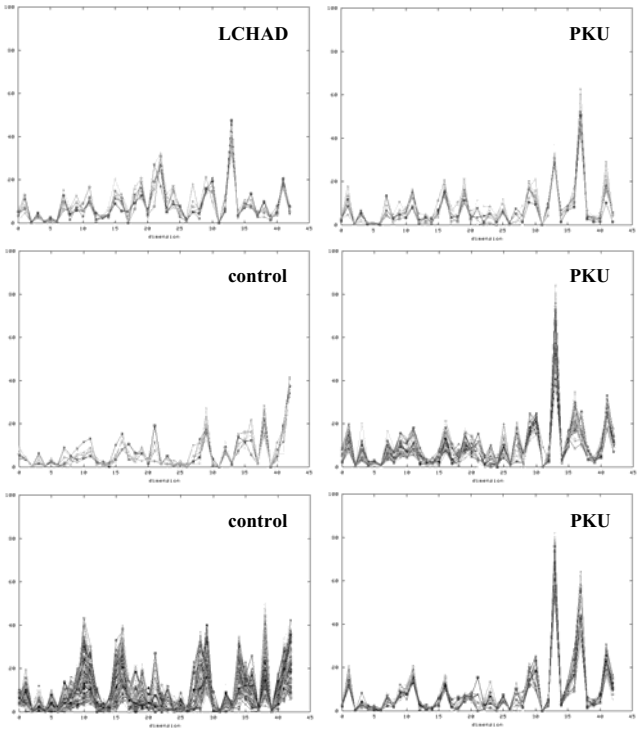


Figure 9: Clusters found by 4C on the metabolome data set. Parameters: $\varepsilon = 150.0$, $\mu = 8$, $\lambda = 20$, $\delta = 0.1$.

6.2.4 *Input Parameters*

The algorithm 4C needs four input parameters which are discussed in the following:

The parameter $\varepsilon \in \mathbb{R}$ specifies the size of the local areas in which the correlations are examined and thus determines the number of objects which contribute to the covariance matrix and consequently to the correlation similarity measure of each object. It also participates in the determination of the density threshold, a cluster must exceed. Its choice usually depends on the volume of the data space (i.e. the maximum value of each attribute and the dimensionality of the feature space). The choice of ε has two aspects. First, it should not be too small because in that case, an insufficient number of objects contribute to the correlation similarity measure of each object and thus, this measure can be meaningless. On the other hand, ε should not be too large because then some noise objects might be correlation reachable from objects within a correlation connected cluster. Let us note, that our experiments indicated that the second aspect is not significant for 4C (in contrast to ORCLUS).

The parameter $\mu \in \mathbb{N}$ specifies the number of neighbors an object must find in an ε-neighborhood and in a correlation ε-neighborhood to exceed the density threshold. It determines the minimum cluster size. The choice of μ should not be to small ($\mu \geq 5$ is a reasonable lower bound) but is rather insensitive in a broad range of values.

Both ε and μ should be chosen hand in hand.

The parameter $\lambda \in \mathbb{N}$ specifies the correlation dimension of the correlation connected clusters to be computed. As discussed above, the correlation dimension of a correlation connected cluster corresponds to its intrinsic dimension. In our experiments, it turned out that λ can be seen as an upper bound for the correlation dimension of the detected correlation connected clusters. However, the computed clusters tend to have a correlation dimension close to λ.

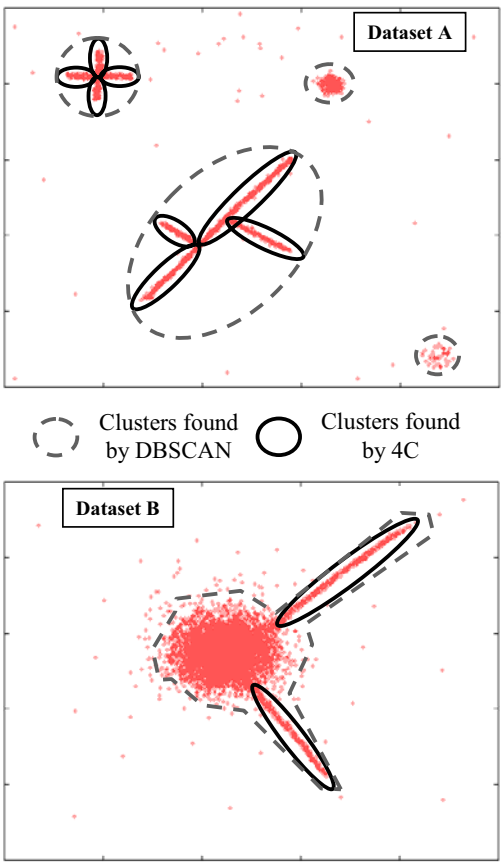


Figure 10: Comparison between 4C and DBSCAN.

The parameter $\delta \in \mathbb{R}$ (where $0 \leq \delta \leq 1$) specifies the lower bound for the decision whether an eigenvalue is set to 1 or to $\kappa \gg 1$. It empirically turned out that the choice of δ influences the tightness of the detected correlations, i.e. how much local variance from the correlation is allowed. Our experiments also showed that $\delta \leq 0.1$ is usually a good choice.

7. CONCLUSIONS

In this paper, we have proposed 4C, an algorithm for computing clusters of correlation connected objects. This algorithm searches for local subgroups of a set of feature vectors with a uniform correlation. Knowing a correlation is potentially useful because a correlation indicates a causal dependency between features. In contrast to well-known methods for the detection of correlations like the principal components analysis (PCA) our algorithm is capable to separate different subgroups in which the dimensionality as well as the type of the correlation (positive/negative) and the dependent features are different, even in the presence of noise points.

Our proposed algorithm 4C is determinate, robust against noise, and efficient with a worst case time complexity between $O(d^2 \cdot n \log n + d^3 \cdot n)$ and $O(d^2 \cdot n^2 + d^3 \cdot n)$. In an extensive evaluation, data from gene expression analysis and

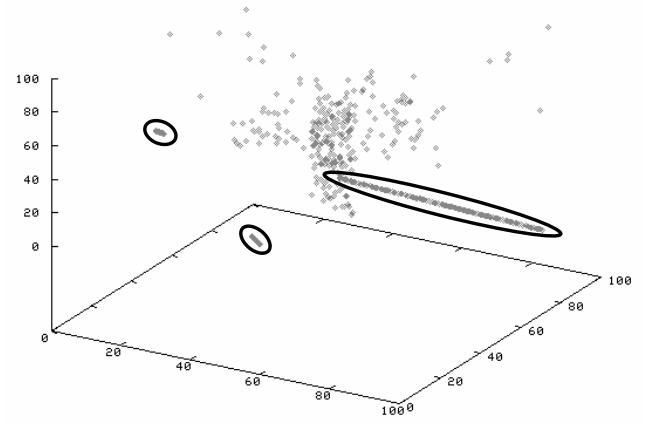


Figure 11: Three correlation connected clusters found by 4C on a 3-dimensional data set. Parameters: $\varepsilon = 2.5$, $\mu = 8$. $\delta = 0.1$, $\lambda = 2$.

metabolic screening have been used. Our experiments show a superior performance of 4C over methods like DBSCAN, CLIQUE, and ORCLUS.

In the future we will integrate our method into object-relational database systems and investigate parallel and distributed versions of 4C.

8. REFERENCES

[1] C. Aggarwal and P. Yu. "Finding Generalized Projected Clusters in High Dimensional Space". In *Proc. ACM SIGMOD Int. Conf. on Management of Data (SIGMOD'00), Dallas, TX*, 2000.

[2] C. C. Aggarwal and C. Procopiuc. "Fast Algorithms for Projected Clustering". In *Proc. ACM SIGMOD Int. Conf. on Management of Data (SIGMOD'99), Philadelphia, PA*, 1999.

[3] R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan. "Automatic Subspace Clustering of High Dimensional Data for Data Mining Applications". In *Proc. ACM SIGMOD Int. Conf. on Management of Data (SIGMOD'98), Seattle, WA*, 1998.

[4] M. Ankerst, M. M. Breunig, H.-P. Kriegel, and J. Sander. "OPTICS: Ordering Points to Identify the Clustering Structure". In *Proc. ACM SIGMOD Int. Conf. on Management of Data (SIGMOD'99), Philadelphia, PA*, 1999.

[5] D. Barbara and P. Chen. "Using the Fractal Dimension to Cluster Datasets". In *Proc. ACM SIGKDD Int. Conf. on Knowledge Discovery in Databases (SIGKDD'00), Boston, MA*, 2000.

[6] S. Berchtold, C. Böhm, H. V. Jagadish, H.-P. Kriegel, and J. Sander. "Independent Quantization: An Index Compression Technique for High-Dimensional Data Spaces". In *Proc. Int. Conf. on Data Engineering (ICDE 2000), San Diego, CA*, 2000.

[7] S. Berchtold, D. A. Keim, and H.-P. Kriegel. "The X-Tree: An Index Structure for High-Dimensional Data". In *Proc. 22nd Int. Conf. on Very Large Databases (VLDB'96), Mumbai (Bombay), India*, 1996.

[8] K. Chakrabarti and S. Mehrotra. "Local Dimensionality Reduction: A New Approach to Indexing High Dimensional Spaces". In *Proc. 26th Int. Conf. on Very Large Databases (VLDB'00), Cairo, Egypt*, 2000.

[9] C.-H. Cheng, A.-C. Fu, and Y. Zhang. "Entropy-Based Subspace Clustering for Mining Numerical Data". In *Proc. ACM SIGKDD Int. Conf. on Knowledge Discovery in Databases (SIGKDD'99), San Diego, FL*, 1999.

[10] 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. 2nd Int. Conf. on Knowledge Discovery and Data Mining (KDD'96), Portland, OR*, 1996.

[11] S. Goil, H. Nagesh, and A. Choudhary. "MAFIA: Efficiant and Scalable Subspace Clustering for Very Large Data Sets". Tech. Report No. CPDC-TR-9906-010, Center for Parallel a´nd Distributed Computing, Dept. of Electrical and

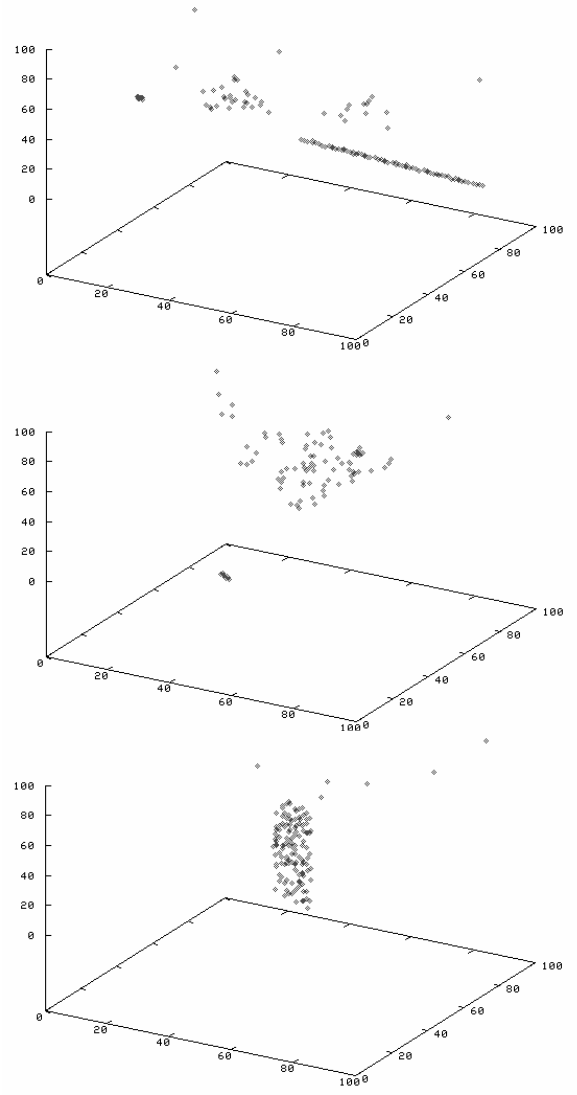


Figure 12: Clusters found by ORCLUS on the data set depicted in Figure 11. Parameters: $k = 3$, $l = 2$.

Computer Engineering, Northwestern University, 1999.

[12] A. Hinneburg and D. A. Keim. "An Efficient Approach to Clustering in Large Multimedia Databases with Noise". In *Proc. 4th Int. Conf. on Knowledge Discovery and Data Mining (KDD'98), New York, NY*, 1998.

[13] B. Liebl, U. Nennstiel-Ratzel, R. von Kries, R. Fingerhut, B. Olgemöller, A. Zapf, and A. A. Roscher. "Very High Compliance in an Expanded MS-MS-Based Newborn Screening Program Despite Written Parental Consent". *Preventive Medicine*, 34(2):127–131, 2002.

[14] J. Liu and W. Wang. "OP-Cluster: Clustering by Tendency in High Dimensional Spaces". In *Proc. of the 3rd IEEE International Conference on Data Mining (ICDM 2003), Melbourne, Fl*, 2003.

[15] E. Parros Machado de Sousa, C. Traina, A. Traina, and C. Faloutsos. "How to Use Fractal Dimension to Find Correlations between Attributes". In *Proc. KDD-Workshop on Fractals and Self-similarity in Data Mining: Issues and Approaches*, 2002.

[16] J. Pei, X. Zhang, M. Cho, H. Wang, and P. S. Yu. "MaPle: A Fast Algorithm for Maximal Pattern-based Clustering". In *Proc. of the 3rd IEEE International Conference on Data Mining (ICDM 2003), Melbourne, Fl*, 2003.

[17] C. M. Procopiuc, M. Jones, P. K. Agarwal, and M. T. M. "(a monte carlo algorithm for fast projective clustering)". In *Proc. ACM SIGMOD Int. Conf. on Management of Data (SIGMOD'02), Madison, Wisconsin*, 2002.

[18] Saccharomyces Genome Database (SGD). http://www.yeastgenome.org/. (visited: Oktober/November 2003).

[19] J. Sander, M. Ester, H.-P. Kriegel, and X. Xu. "Density-Based Clustering in Spatial Databases: The Algorithm GDBSCAN and its Applications". *Data Mining and Knowledge Discovery*, 2:169–194, 1998.

[20] S. Tavazoie, J. D. Hughes, M. J. Campbell, R. J. Cho, and C. G. M. "Systematic Determination of Genetic Network Architecture". *Nature Genetics*, 22:281–285, 1999.

[21] H. Wang, W. Wang, J. Yang, and P. S. Yu. "Clustering by Pattern Similarity in Large Data Set". In *Proc. ACM SIGMOD Int. Conf. on Management of Data (SIGMOD'02), Madison, Wisconsin*, 2002.

[22] X. Xu, M. Ester, H.-P. Kriegel, and J. Sander. "A Distribution-Based Clustering Algorithm for Mining in Large Spatial Databases". In *Proc. 14th Int. Conf. on Data Engineering (ICDE'98), Orlando, FL*, pages 324–331. AAAI Press, 1998.

[23] J. Yang, W. Wang, H. Wang, and P. S. Yu. "Delta-Clusters: Capturing Subspace Correlation in a Large Data Set". In *Proc. Int. Conf. on Data Engineering (ICDE 2002), San Jose, CA*, 2002.