Clustering High-Dimensional Data: A Survey on Subspace Clustering, Pattern-Based Clustering, and Correlation Clustering

HANS-PETER KRIEGEL, PEER KRÖGER, and ARTHUR ZIMEK
Ludwig-Maximilians-Universität München

As a prolific research area in data mining, subspace clustering and related problems induced a vast quantity of proposed solutions. However, many publications compare a new proposition—if at all—with one or two competitors, or even with a so-called "naïve" ad hoc solution, but fail to clarify the exact problem definition. As a consequence, even if two solutions are thoroughly compared experimentally, it will often remain unclear whether both solutions tackle the same problem or, if they do, whether they agree in certain tacit assumptions and how such assumptions may influence the outcome of an algorithm. In this survey, we try to clarify: (i) the different problem definitions related to subspace clustering in general; (ii) the specific difficulties encountered in this field of research; (iii) the varying assumptions, heuristics, and intuitions forming the basis of different approaches; and (iv) how several prominent solutions tackle different problems.

Categories and Subject Descriptors: A.1 [**General Literature**]: Introductory and Survey; H.2.8 [**Database Management**]: Database Applications—*Data mining*; I.5.3 [**Pattern Recognition**]: Clustering

General Terms: Algorithms, Theory

Additional Key Words and Phrases: Survey, clustering, high-dimensional data

ACM Reference Format:

Kriegel, H.-P., Kröger, P., and Zimek, A. 2009. Clustering high-dimensional data: A survey on subspace clustering, pattern-based clustering, and correlation clustering. ACM Trans. Knowl. Discov. Data. 3, 1, Article 1 (March 2009), 58 pages. DOI = 10.1145/1497577.1497578 http://doi.acm.org/10.1145/1497577.1497578

1. INTRODUCTION

Clustering aims at dividing datasets into subsets (clusters), where objects in the same subset are similar to each other with respect to a given similarity measure, whereas objects in different clusters are dissimilar. Cluster analysis has

Author's address: H.-P. Kriegel, P. Kröger, and A. Zimek, Institute for Informatics, Ludwig-Maximilians-Universität München, Oettingenstr. 67, 80538 Munich, Germany; email: {kriegel, kroegerp,zimek}@dbs.ifi.lmu.de.

DOI 10.1145/1497577.1497578 http://doi.acm.org/10.1145/1497577.1497578

been used in a large variety of fields, such as astronomy, physics, medicine, biology, archaeology, geology, geography, psychology, and marketing. Many different research areas contributed new approaches (i.e., pattern recognition, statistics, information retrieval, machine learning, bioinformatics, and data mining). In some cases, the goal of cluster analysis is a better understanding of the data (e.g., learning the "natural" structure of data which should be reflected by a meaningful clustering). In other cases, cluster analysis is merely a first step for different purposes, such as indexing or data compression.

While clustering in general is a rather dignified problem, mainly in about the last decade new approaches have been proposed to cope with new challenges provided by modern capabilities of automatic data generation and acquisition in more and more applications, producing a vast amount of high-dimensional data. This data needs to be analyzed by data mining methods in order to gain the full potential from the gathered information. However, high-dimensional data poses different challenges for clustering algorithms that require specialized solutions. In particular, in high-dimensional data, traditional similarity measures as used in conventional clustering algorithms are usually not meaningful, as we will discuss in what follows. This problem and related phenomena require adaptations of clustering approaches to the nature of high-dimensional data. Common approaches are known as, for example, subspace clustering, projected clustering, pattern-based clustering, or correlation clustering. This area of research has been a highly active one in recent years, with a plethora of proposed algorithms; however, in our opinion, there has been a lack of a systematic problem analysis. Thus, a comparison of proposed algorithms is difficult, both theoretically and practically.

Since almost all research on specialized approaches to clustering high-dimensional data is relatively new, it is not covered in most textbooks in different related fields (data mining, statistics, machine learning, pattern recognition; refer to, e.g., Hand et al. [2001], Hastie et al. [2001], Witten and Frank [2005], Bishop [2006]) or in not-so-recent surveys on the topic of clustering [Jain et al. 1999]. Others [Duda et al. 2001; Han and Kamber 2001; Tan et al. 2006] sketch the problem rather casually. A more recent edition [Han and Kamber 2006] at least dedicates a section to the problem, sketching some example algorithms and touching on some problems.

Recently, some surveys have given overviews on some approaches. In the well-known survey of Parsons et al. [2004] the problem is introduced in a very illustrative way and some approaches are sketched. However, there is no clear distinction between different subproblems (axis-parallel or arbitrarily oriented) and the corresponding algorithms are discussed without pointing out the underlying differences in the respective problem definitions. Van Mechelen et al. [2004] gave an overview on older, more specialized work of a special type of pattern-based clustering (biclustering) in the medical and biological area from a statistics point of view. Madeira and Oliveira [2004] focus on pattern-based clustering approaches and are especially interested in the application domain of microarray data. Jiang et al. [2004] focus exclusively on the application domain of gene expression data and discuss clustering approaches structured according to the application scenarios. However, in addition to the applications

of full-dimensional clustering approaches, they sketch only three biclustering approaches (named "subspace clustering" in their overview). Tanay et al. [2006] discuss some biclustering algorithms as representatives of different algorithmic approaches, also focused on the application to gene expression data.

Here, we would like to give a more systematic approach to the problem and focus on the different tasks and subproblems (axis-parallel, pattern-based, arbitrarily oriented clustering). Therefore, we will also survey the related heuristics used by different approaches. Our systematic view is not based on the application scenarios, but on the intrinsic methodological differences of the various families of approaches based on different spatial intuitions. Thus, we will also try to integrate the inherently different point of view of pattern-based approaches into the intuition of patterns in data space.

In this introductory section, we first give an overview on some sample applications where high-dimensional data occur (Section 1.1). Then in Section 1.2, we shortly sketch and distinguish the general problems. Finally, we give the outline of the survey in Section 1.3.

1.1 Sample Applications of Clustering High-Dimensional Data

In many applications, cluster analysis of high-dimensional data is very important. Here, four probably rather prominent examples are described.

1.1.1 *Gene Expression Analysis.* Microarray chip technology produces a large amount of data in molecular biology. Microarray data (also called gene expression data) contains the "expression level" of genes measured under different conditions (e.g., in different tissues) under varying experimental environments, at different time slots after special treatment, or from different test persons. The expression level of a gene allows to draw conclusions about the amount of the corresponding gene product, such as a protein or a regulatory RNA, in the particular cell. Microarray data usually comprises the simultaneous measurement of the expression level of thousands of genes under hundreds of conditions. It consists of a data matrix where the rows correspond to genes and the columns represent different experimental conditions, different tissues, consecutive time slots, or different test persons. Biologists usually want to find patterns in such massive datasets. Depending on the scope of their research, the data mining task can vary.

—*Clustering the Rows.* Very often, the biologists want to find groups of genes with homogeneous expression levels indicating that these genes share a common function. In this case, the columns usually represent different experimental conditions or different time slots within a time-dependent experiment. In general, genes may have very different functions depending on the cellular environment. Thus, the genes will usually cluster differently in varying subsets of the conditions or time slots. In other words, the clustering algorithm should take into account that a given gene A may, for example, be grouped together with gene B but not with gene C in a subset S of the columns, whereas A may, for example, be grouped together with C but not with B in another subset T of columns.

—*Clustering the Columns.* In medical research, microarray data is often used to find genetic relationships and disorders. In this case, the columns of the gene expression data matrix represent different individuals. The data mining task is to cluster these individuals. Any clustering algorithm has to take into account that the individuals usually differ in many phenotypical aspects (e.g., gender, age, hair color, specific diseases, etc.). These different phenotypes are caused by different subsets of genes. In other words, for example, the individual A may be grouped together with individual B but not with individual C in a subset S of genes, whereas A may be grouped together with C but not with B in another subset T of genes.

—*Coclustering Rows and Columns.* Especially in this application domain, the combination of both these tasks, that is, simultaneously clustering the rows and the columns of a gene expression data matrix, is considered a specialized problem and a special family of algorithms is dedicated to solve it. However, the results of clustering rows and clustering columns can be translated into one another and both pose a classical problem description for subspace clustering.

1.1.2 *Metabolic Screening.* Many governments have implemented a metabolic screening of newborns in order to detect metabolic diseases in the earliest possible moment. For this purpose, a blood sample is taken from each newborn and the concentrations of specific metabolites (i.e., metabolic products) in these blood samples are measured. In the resulting data matrix, rows represent the newborns and columns represent the metabolites. Biologists usually want to identify homogeneous groups of newborns suffering from a common metabolic disease. Usually, each metabolic disease causes a correlation between the concentration of a specific set of metabolites. Thus, any clustering algorithm should take into account that newborns should be grouped together only if they exhibit a common correlation among a set of metabolites. In addition, the set of participating metabolites and the type of correlation can be different for different diseases (i.e., clusters).

1.1.3 *Customer Recommendation Systems.* In customer recommendation systems, customers of a company can vote for the company's products. Depending on the portfolio of the company, there may be a very large set of products. It is now interesting, for example, for target marketing purposes, to cluster the customers into groups of homogeneous voting schemata. Customers that have similar preferences should be grouped together. For each group, special marketing strategies can be applied taking each group's preferences into account. The problem for a cluster analysis process is that different customers may be grouped together according to different sets of products. In other words, customer A may share a preference for a given set S of products with customer B but not with C, whereas A may share another preference for a different set T of products with C but not with B. To make the problem even more challenging, the relationships between the preferences of the customers of one cluster may be arbitrary complex, like "The lower the products p_1 and p_2 are rated, the higher the products p_3 and p_4 are rated." Symmetrically, clustering products based on similar customer preferences (where only a small subset of customers

may be sufficient to establish similarity among products) is also a common problem.

1.1.4 *Text Documents.* Clustering collections of text documents such as Web pages to find groups of thematically related documents is important in many applications. Usually, the text documents are transformed into high-dimensional feature vectors, such as when using term frequency features. The data matrix to be analyzed then contains each document as a row where the columns represent the count of one particular term in the corresponding document. Since the count for any term occurring in any text document (after excluding stop-words, applying stemming, etc.) needs to be recorded, usually the data contains thousands of attributes and is thus very sparse, featuring a lot of zero values. Again, related documents will only have a similar word count in a subset of terms, and these subsets are likely to be different for different groups of thematically relevant documents. In addition, the thematic groups may overlap, that is, one document may be assigned to more than one thematic group according to similarities of the count value in different subsets of terms. In other words, document A may share a similar frequency in a given set S of terms with document B but not with C, whereas A may share another similar frequency in a different set T of terms with C but not with B.

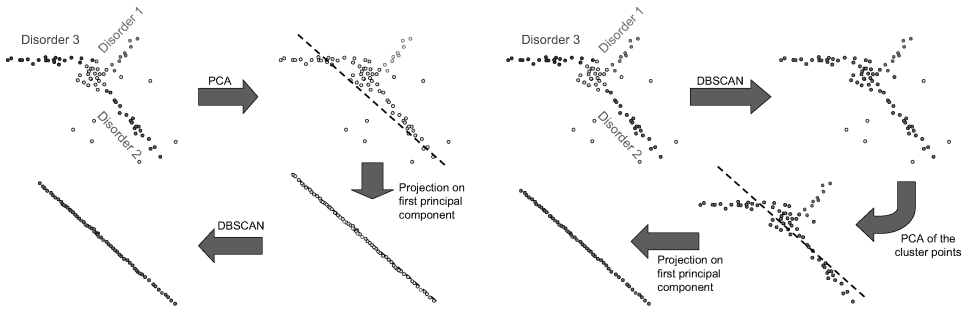
Note that in the area of text analysis, often specialized distance functions (e.g., disregarding zero values for the distance computation) are used. In this survey, we will not discuss specialized distance functions in this sense. Nevertheless, the general problem statement of this application area is also a typical illustration for the special requirements of clustering high-dimensional data.

1.2 Finding Clusters in High-Dimensional Data

As sketched in the previous application scenarios, when clustering high-dimensional data, we face different problems. The presence of irrelevant features or of correlations among subsets of features heavily influences the appearance of clusters in the full-dimensional space. The main challenge for clustering here is that different subsets of features are relevant for different clusters, that is, the objects cluster in subspaces of the data space but the subspaces of the clusters may vary. Additionally, different correlations among the attributes may be relevant for different clusters. We call this phenomenon that different features or a different correlation of features may be relevant for varying clusters the *local feature relevance* or *local feature correlation*.

A common way to overcome problems of high-dimensional data spaces where several features are correlated or only some features are relevant is to perform feature selection before performing any other data mining task. Feature selection methods like Principal Component Analysis (PCA) can be used to map the original data space to a lower-dimensional data space where the points may cluster better and the resulting clusters may be more meaningful.

Unfortunately, such feature selection or dimensionality reduction techniques cannot be applied to clustering problems. Feature selection or dimensionality reduction techniques are *global* in the following sense: They generally compute only one subspace of the original data space in which the clustering can then be



(a) first dimensionality reduction then clustering (b) first clustering then dimensionality reduction

Fig. 1. Illustration of the local feature relevance/local feature correlation problem.

performed, considering the complete set of points. In contrast, the problem of *local feature relevance* and *local feature correlation* states that multiple subspaces are needed because each cluster may exist in a different subspace. Figure 1(a) illustrates this problem for a fictive sample two-dimensional dataset derived from the metabolic screening application. The data contains a set of patients, some of them healthy, others suffering from specific metabolic diseases. For each patient, the concentrations of two fictive metabolites are measured. There are four clusters (healthy, disorder 1–3) and some noise. Whereas the cluster of the healthy patients forms a conventional two-dimensional cluster, for each cluster representing ill patients different feature relevance and feature correlation applies: For disorder 1, a positive correlation between both attributes can be observed. For disorder 2, a negative correlation between both attributes can be observed. For disorder 3, only the feature represented by the y-axis is relevant. If a feature selection method (here PCA) is applied to this data in order to reduce the dimensionality by one, the four clusters cannot be separated in the resulting subspace anymore (refer to Figure 1(a)). On the other hand, if the points in the original data space are clustered first (here using DBSCAN [Ester et al. 1996]) and afterwards feature selection is applied to each resulting cluster (e.g., again using PCA), also no reasonable result can be detected (refer to Figure 1(b)). In summary, due to the problem of local feature relevance and local feature correlation, usually no global feature selection can be applied to overcome the challenges of clustering high-dimensional data.

Instead of a global approach to feature selection, a local approach accounting for the local feature relevance and/or local feature correlation problems is required. Since traditional methods, like feature selection, dimensionality reduction, and conventional clustering, do obviously not solve the previously sketched problems, novel methods need to integrate feature analysis into the clustering process more tightly. Figure 2 illustrates the general challenge for finding clusters in high-dimensional data. Cluster 3 exists in an axis-parallel subspace, clusters 1 and 2 exist in (different) arbitrarily oriented subspaces: If the cluster members are projected onto the depicted subspaces, the points are densely packed (i.e., are similar to each other). Generally, we can derive the following aim for methods that are designed for clustering high-dimensional

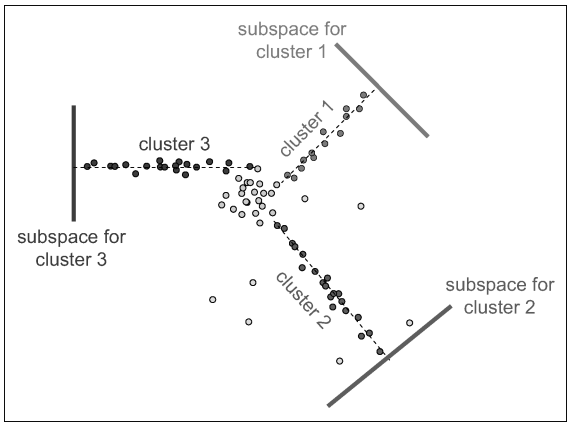


Fig. 2. Illustration of the general aim of clustering algorithms for high-dimensional data.

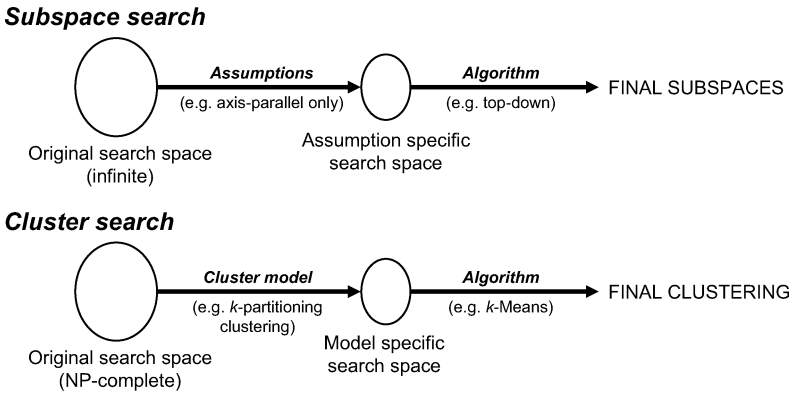


Fig. 3. Illustration of the two general problems of clustering high-dimensional data.

data:

> The general aim of clustering algorithms designed for high-dimensional data is to find clusters in arbitrarily oriented subspaces of the original feature space.

Of course, the meaning of the term "clusters" is still open to debate. In fact, when clustering high-dimensional data, we face two separate problems. The first problem is the search for the relevant subspaces and the second problem focuses on the detection of the final clusters. The search space for the subspace search problem in its general form is tentatively infinite. The most general form of the clustering problem is known to be NP-complete [Slagle et al. 1975]. As mentioned before, both problems need to be solved simultaneously. Since both problems feature a very challenging search space, we need to employ heuristics for both problems to develop such solutions. Figure 3 illustrates these considerations. While many heuristics and assumptions for efficiently solving the clustering problem have been discussed in the past decades, we will focus on

the subspace search problem in this survey. However, for the sake of completeness, we should keep in mind the fact that the solutions discussed here also rely on assumptions and heuristics that consider the clustering problem. These assumptions and heuristics are reflected by the underlying clustering model that specifies the meaning of clusters and the algorithms to compute such clusters. One important aspect to describe the meaning of clusters that has an impact on the subspace search problem is the choice of a certain distance or similarity measure. Which data points are considered more similar than others, and thus cluster together, crucially depends on this choice. Some approaches are based on neighborhood in Euclidean space, typically using an L_p-norm as distance measure. Others take into account a similar "behavior" of attribute values over a certain range of attributes (often called *patterns*). But different approaches also make use of the definition of similarity in different ways, based on different intuitions of the meaning of a cluster and resulting in different cluster models.

Since there are different approaches to define a "cluster" in general, there are also different notions of what constitutes a "subspace cluster". This survey aims at discussing different assumptions concerning the subspace search problem, different ways to grasp "similarity" among data points, and, overall, different general algorithmic approaches to finding clusters in high-dimensional data.

1.3 Outline

A naïve solution to the general aim of clustering algorithms for high-dimensional data is to test all possible arbitrarily oriented subspaces for clusters. Obviously, there is an infinite number of arbitrarily oriented subspaces, so this naïve solution is computationally infeasible. Rather, we need some heuristics and assumptions in order to conquer this infinite search space.

As Figure 2 suggests, the clusters can in general be found in arbitrarily oriented subspaces. However, in some applications, it is reasonable to focus only on clusters in axis-parallel subspaces (like cluster 3 in Figure 2). In this case, the search space of all potential subspaces accommodating clusters is restricted, but still in $O(2^d)$. Many algorithms proposed to-date use this restriction and are limited to finding clusters in axis-parallel subspaces only. In the literature, these clustering algorithms are usually called *projected clustering* or *subspace clustering* algorithms. We review and discuss the problem of finding clusters in axis-parallel subspaces in Section 2.

On the other hand, several applications require solutions for the general case where clusters may exist in any arbitrarily oriented subspace. The algorithms of this class of solutions are usually called correlation clustering algorithms. Let us note that some authors use the term "subspace clustering algorithm" interchangeably also with "correlation clustering algorithm." We review and discuss the problem of finding clusters in arbitrarily oriented subspaces in Section 4.

In between these two main classes of existing algorithms, namely finding clusters in axis-parallel subspaces and clusters in arbitrarily oriented subspaces, a third class of algorithms following a slightly different approach has been proposed. Those algorithms are typically referred to as pattern-based clustering (or sometimes biclustering, coclustering) algorithms. In fact, some

pattern-based clustering algorithms are restricted to axis-parallel subspace clusters, whereas other pattern-based clustering algorithms are not restricted to axis-parallel subspace clusters but are limited to clusters in special cases of arbitrarily oriented subspaces. However, since pattern-based clustering methods take a different approach to the problems presented here and expose a kind of hybrid approach between axis-parallel and arbitrarily oriented subspace clustering, we discuss them as a separate class of algorithms in Section 3.

Having introduced the different concepts and approaches, we will discuss the approaches again in a more general way comparing the different problem statements as well as the heuristics and the related restrictions of the different approaches (refer to Section 5). We summarize this survey in Section 6 and share some thoughts regarding future work in this area.

2. FINDING CLUSTERS IN AXIS-PARALLEL SUBSPACES

A very common assumption to shrink down the infinite search space of all possible subspaces is to focus on axis-parallel subspaces only. The assumption that clusters can only be found in axis-parallel subspaces may be rather sensible in the context of various applications. The big advantage is that the search space is now restricted by the number of all possible axis-parallel subspaces. However, the bound is still rather high: In a d-dimensional dataset the number of k-dimensional subspaces is $\binom{d}{k}$ $(1 \leq k \leq d)$, and thus the number of all possible subspaces is

$$\sum_{k=1}^{d} \binom{d}{k} = 2^d - 1.$$

In the literature, the problem of finding axis-parallel clusters has been referred to as "projected clustering" and "subspace clustering." However, these terms are not consistently used in the literature, causing some potential misunderstandings. This article is also an attempt to establish a standard vocabulary. Originally, projected clustering and subspace clustering refer to two different subproblems of finding clusters in axis-parallel subspaces (or projections). However, in the literature, these two subproblems have been mixed up, for example, with the algorithmic approach used to conquer the search space of possible subspaces to look for clusters. For example, the distinction between "dimension-growth subspace clustering" and "dimension-reduction projected clustering" given by Han and Kamber [2006] is such a mix-up. In the following, we give two classification schemata of existing algorithms to find clusters in axis-parallel subspaces. The first schema is a problem-oriented view of the task and ends up in defining the terms "projected clustering" and "subspace clustering" in a unified way. The second schema is an algorithmic view, analyzing the algorithmic approach employed to conquer the exponential search space of possibly interesting subspaces.

2.1 A Problem-Oriented Categorization

Due to the aforementioned exponential search space, all algorithms that are limited to finding clusters in axis-parallel subspaces rely on further

assumptions that usually affect the results produced. In the literature, there are generally four different classes of problem statements, depending on the assumptions made according to which the algorithms can be classified. Note, however, that problem statements are often not stated explicitly.

(1) *Projected Clustering Algorithms*. A first class of algorithms aims at finding a unique assignment of each point to exactly one subspace cluster (or noise). Generally, these try to find the projection where the currently considered set of points clusters best. These algorithms are referred to as *projected clustering* algorithms.
(2) *"Soft" Projected Clustering Algorithms*. Some projected clustering algorithms further assume that the number k of clusters is known beforehand such that an objective function can be defined which is optimized to derive the optimal set of k clusters. In these cases, however, usually the subspaces are not assigned in a "hard" way to the clusters. Different attributes are just differently weighted but all attributes contribute to the clustering. Thus, there is a group of approaches known as *"soft" projected clustering*. This group can also be seen as a subclass of projected clustering algorithms.
(3) *Subspace Clustering Algorithms*. A third class of algorithms aims at finding all subspaces where clusters can be identified. Thus, these algorithms are dedicated to finding all clusters in all subspaces. We refer to this group of algorithms as *subspace clustering* algorithms.
(4) *Hybrid Algorithms*. A fourth class of algorithms aims at finding something in between. Usually, these algorithms aim at finding clusters that may overlap. On the other hand, these algorithms do not aim at finding *all* clusters in *all* subspaces. Some of the hybrid algorithms only compute interesting subspaces rather than final subspace clusters. The reported subspaces can then be mined by applying full-dimensional algorithms to these projections.

Let us note that all classes of algorithms imply that there is a definition of what constitutes a cluster. The output of these algorithms is a list of clusters, each represented as a pair (X, Y), where X is a subset of data objects and Y is a subset of data attributes, such that the points in X meet a given cluster criterion when projected onto the attributes in Y but do not meet the cluster criterion when projected onto the remaining attributes (i.e., the points in X are "close" when projected onto the attributes in Y but projected onto the remaining attributes they are "not close"). Usually, the cluster criterion and the measure of closeness differs from algorithm to algorithm. Recently, an attempt to define what constitutes a subspace cluster in a statistically sound way has been described by Moise and Sander [2008].

2.2 An Algorithmic-Oriented Categorization

A second classification schema of existing algorithms for finding clusters in axis-parallel subspaces focuses on the algorithmic approach to conquer the exponential search space of all possible subspaces. In general, this is an important view because efficiently navigating through this search space is one of the key challenges for the design of an axis-parallel projected, subspace, or

hybrid clustering algorithm. The task is to efficiently identify those subspaces that accommodate one or more clusters. Compared to evaluating a given cluster criterion, the search for the subspaces accommodating a cluster is usually the bottleneck, even for low-dimensional data. For example, for $d = 20$, we face more than 1 million possible subspaces. A complete enumeration of these subspaces is obviously computationally infeasible.

In general, the algorithmic approaches for finding these subspaces (i.e., traversing the search space of all possible axis-parallel subspaces) can be divided into the following two categories.

(1) *Top-Down Approaches*. The rationale of top-down approaches is to determine the subspace of a cluster starting from the full-dimensional space. This is usually done by determining a subset of attributes for a given set of points (potential cluster members) such that the points meet the given cluster criterion when projected onto the corresponding subspace. Obviously, the dilemma is that for the determination of the subspace of a cluster, at least some cluster members must be identified. On the other hand, in order to determine cluster memberships, the subspace of each cluster must be known. To escape from this circular dependency, most top-down approaches rely on a rather strict assumption, which we call the *locality assumption*. It is assumed that the subspace of a cluster can be derived from the local neighborhood (in the full-dimensional data space) of the cluster center or the cluster members. In other words, it is assumed that even in the full-dimensional space, the subspace of each cluster can be learned from the local neighborhood of cluster representatives or cluster members. Other top-down approaches that do not rely on the locality assumption use random sampling in order to generate a set of potential cluster members.

(2) *Bottom-Up Approaches*. The exponential search space that needs to be traversed is equivalent to the search space of the frequent item set problem in market basket analysis in transaction databases [Agrawal and Srikant 1994]. Each attribute represents an item and each subspace cluster is a transaction of the items representing the attributes that span the corresponding subspace. Finding itemsets with frequency 1 then relates to finding all combinations of attributes that constitute a subspace containing at least one cluster. This observation is the rationale of most bottom-up subspace clustering approaches. The subspaces that contain clusters are determined starting from all one-dimensional subspaces that accommodate at least one cluster by employing a search strategy similar to frequent item set mining algorithms. To apply any efficient frequent item set mining algorithm, the cluster criterion must implement a downward closure property (also called monotonicity property): *If subspace S contains a cluster, then any subspace $T \subseteq S$ must also contain a cluster.* The reverse implication, *if a subspace T does not contain a cluster, then any superspace $S \supseteq T$ also cannot contain a cluster*, can be used for pruning, that is, excluding specific subspaces from consideration. Let us note that there are bottom-up algorithms that do not use an APRIORI-like subspace search, but instead apply other search heuristics.

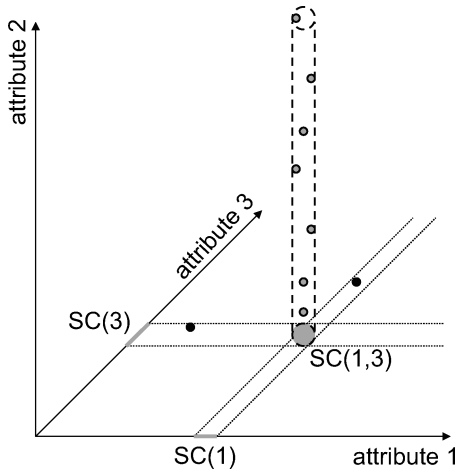


Fig. 4. Illustration of the general approaches to find axis-parallel subspace clusters.

Both the top-down and bottom-up approaches are commonly used in the literature. While the top-down approach tries to anticipate cluster members and then determines the subspace of each cluster, the bottom-up approach instead tries to anticipate the subspaces of the clusters and then determines the cluster members. These rationales are depicted in Figure 4. Most of the points of a three-dimensional data space form a subspace cluster with relevant attributes 1 and 3 because these points show low variance along these attributes. In other words, when projected on the subspace spanned by attributes 1 and 3, these points are densely packed. This subspace cluster is referred to as $SC(1, 3)$ in the figure. Along attribute 2 the variance of the points is high. Bottom-up approaches explore this cluster from its one-dimensional projections $SC(1)$ and $SC(3)$. First, these projections are computed and then, if special conditions hold, these projections are merged. Note that here two points that do not belong to $SC(1, 3)$ belong to $SC(1)$ or $SC(3)$. Thus, $SC(1)$ and $SC(3)$ differ from $SC(1, 3)$ not only with respect to the attributes that are considered relevant, but also with respect to their corresponding cluster members. Top-down approaches would start with a cluster member or the cluster center (assuming that the initial cluster center is near the correct cluster center in the full-dimensional space) and would compute its neighborhood in the full three-dimensional space. Then, the variance of the neighbors is usually examined along each dimension in order to determine the correct subspace of a cluster. In this case, the attributes 1 and 3 will most likely be considered as relevant, that is, the cluster $SC(1, 3)$ will be found. On the other hand, a top-down approach is most likely not able to find the projections $SC(1)$ and $SC(3)$.

2.3 Survey and Categorization of Existing Approaches

In the following, we survey representative solutions, categorized according to the task definition they adopt.

2.3.1 *Projected Clustering Algorithms.* Projected clustering algorithms aim at finding a unique assignment of points to subspace clusters. Some algorithms also model noise explicitly, that is, points are assigned uniquely to only one cluster or the noise set. The general approach of these algorithms is to integrate a specialized distance function into a traditional, full-dimensional clustering algorithm. The distance function is usually designed to reflect the subspaces of the true clusters.

PROCLUS [Aggarwal et al. 1999] is a k-medoid-like clustering algorithm. It randomly determines a set of potential cluster centers M on a sample of points first. In the iterative cluster refinement phase, for each of the k current medoids the subspace is determined by minimizing the standard deviation of the distances of the points in the neighborhood of the medoids to the corresponding medoid along each dimension. Points are then assigned to the closest medoid considering the relevant subspace of each medoid. The clusters are refined by replacing bad medoids with new medoids from M, as long as the clustering quality increases. A postprocessing step identifies noise points that are too far away from their closest medoids. The algorithm always outputs a partition of the data points into k clusters (each represented by its medoid) with corresponding subspaces and a (potentially empty) set of noise points. The k-medoid-style cluster model tends to produce equally sized clusters that have spherical shape in their corresponding subspaces. In addition, since the set M of possible medoids is determined in a randomized procedure, different runs of PROCLUS with the same parametrization usually result in different clusterings. Variations of PROCLUS are FINDIT [Woo et al. 2004], employing additional heuristics to enhance efficiency and clustering accuracy, and SSPC [Yip et al. 2005] that offers the capability of further enhancing accuracy by using domain knowledge in the form of labeled objects and/or labeled attributes.

PreDeCon [Böhm et al. 2004] applies the density-based full-dimensional clustering algorithm DBSCAN [Ester et al. 1996] using a specialized distance measure that captures the subspace of each cluster. The definition of this specialized subspace distance is based on the so-called subspace preference that is assigned to each point $\vec{p}$, representing the maximal-dimensional subspace in which $\vec{p}$ clusters best. A dimension is considered relevant for the subspace preference of a point $\vec{p}$ if the variance of points in the Euclidean ε-neighborhood of $\vec{p}$ is below a user-defined threshold δ. The specialized subspace distance between points is a weighted Euclidean distance where the dimensions relevant for the subspace preference of a point are weighted by a constant $\kappa \gg 1$ while the remaining dimensions are weighted by 1. PreDeCon determines the number of clusters automatically, and handles noise implicitly. In addition, its results are determinate and the clusters may exhibit any shape and size in the corresponding subspace. However, PreDeCon requires the user to specify a number of input parameters that are usually hard to guess.

CLTree [Liu et al. 2000] is a method that presents an interesting variation of the theme. The basic idea is to assign a common class label to all existing points and to add additional points uniformly distributed over the data space and labeled as a different class. Then a decision tree is trained to separate the two classes. As a consequence, the attributes are split independently, adaptively,

and in a flexible order of the attributes. However, selecting a split is based on the evaluation of information gain, which is rather costly. Furthermore, the density of the superimposed artificial data can be expected to heavily influence the quality of the results. Since the distribution parameters of existing clusters are unknown beforehand, finding a suitable parametrization seems rather hard. Another problem is the merging of adjacent regions. A cluster can easily become separated if the corresponding bins do not "touch" each other.

2.3.2 *"Soft"-Projected Clustering Algorithms.* Recently, quite a lot of work has been dedicated to developing weighting schemes for k-means-like approaches, in many cases explicitly claiming to address the problems of clustering high-dimensional data. However, if the weighting schemes do not allow to assign a weight $w_i = 0$ to some attributes a_i but all weights remain larger than 0, the problem of irrelevant attributes is not really accounted for. This is the case in many of these approaches. This work is usually related to the statistics and machine learning communities. Accordingly, the approaches are dedicated to maintain the possibility of optimizing an objective function during the clustering process.

Geometrically, the effect of these approaches is just to allow the shape of clusters to become axis-parallel ellipsoids instead of spheres. This does not necessarily account for relevance/irrelevance of different attributes but is more related to some sort of normalization of attributes per cluster. These approaches are sometimes referred to as *soft-projected clustering* algorithms, since they do not assign specific (hard) subspaces to the clusters.

An example for these methods is LAC (Locally Adaptive Clustering) [Domeniconi et al. 2004], which starts with k centroids and k sets of d weights (for d attributes). The algorithm proceeds to approximate a set of k Gaussians by adapting the weights. Cheng et al. [2008] proposed another weighted k-means approach very similar to LAC, but allowing for incorporation of further constraints.

As another prominent example, COSA [Friedman and Meulman 2004] does not derive a clustering but merely a similarity matrix that can be used by an arbitrary clustering algorithm afterwards. The matrix contains weights for each point specifying a subspace preference of the points, similar to PreDeCon. The weights for a point $\vec{p}$ are determined by starting with the Euclidean k-nearest neighbors of $\vec{p}$ and by computing the average distance distribution of the k-nearest neighbors along each dimension. Roughly speaking, the weight for a given attribute is computed as the ratio between the distances of the point $\vec{p}$ to all k-nearest neighbors of $\vec{p}$ in that attribute and the average distances of $\vec{p}$ to all k-nearest neighbors of $\vec{p}$ in all attributes. Thus, the higher the variance of the k-nearest neighbors along an attribute, the higher the weight that is assigned to this attribute. The weight, however, cannot become 0. As long as the weight vectors still change, the k-nearest neighbors are again determined using the current weights and the weights are recomputed. The number of neighbors k is an input parameter. Very different to PreDeCon, the weights can have arbitrary values rather than only two fixed values. In addition, Friedman and Meulman [2004] test the weighting matrix using several full-dimensional

clustering algorithms rather than integrating it into only one specific algorithm. Note that the effect of weight vectors is very different in density-based approaches like PreDeCon where the shape of clusters is not defined beforehand.

As we have seen, these approaches miss the complex interrelation of different problems in high-dimensional data. For further references, the reader may therefore refer to some other examples of recent work in this direction. Although the connections and differences to the algorithms surveyed in the other sections of this survey are not made clear, we found Huang et al. [2005], Bouveyron et al. [2007], and Jing et al. [2007] to provide detailed sections on related work suitable as introduction to the history of soft-projected clustering approaches.

2.3.3 *Subspace Clustering Algorithms.* Subspace clustering algorithms aim at finding all clusters in all subspaces of the entire feature space.

CLIQUE [Agrawal et al. 1998], the pioneering approach to subspace clustering, uses a grid-based clustering notion. The data space is partitioned by an axis-parallel grid into equisized units of width ξ. Only units which contain at least τ points are considered as dense. A cluster is defined as a maximal set of adjacent dense units. Since dense units satisfy the downward closure property, subspace clusters can be explored rather efficiently in a bottom-up way. Starting with all one-dimensional dense units, $(k+1)$-dimensional dense units are computed from the set of k-dimensional dense units in an APRIORI-like style. If a $(k+1)$-dimensional unit contains a projection onto a k-dimensional unit that is not dense, then the $(k+1)$-dimensional unit also can not be dense. Furthermore, a heuristic that is based on the minimum description length principle is introduced to discard candidate units within less interesting subspaces (i.e., subspaces that contain only a very small number of dense units). This way, the efficiency of the algorithm is enhanced but at the cost of incomplete results, namely some true clusters are lost. There are some variants of CLIQUE. The method ENCLUS [Cheng et al. 1999] also relies on a fixed grid, but searches for subspaces that potentially contain one or more clusters rather than for dense units. Three quality criteria for subspaces are introduced, one implementing the downward closure property. The method MAFIA [Nagesh et al. 2001] uses an adaptive grid. The generation of subspace clusters is similar to CLIQUE. Another variant of CLIQUE, called nCluster [Liu et al. 2007], allows overlapping windows of length δ as one-dimensional units of the grid. In summary, all grid-based methods use a simple but rather efficient cluster model. The shape of each resulting cluster corresponds to a polygon with axis-parallel lines in the corresponding subspace. Obviously, the accuracy and efficiency of CLIQUE and its variants primarily depend on the granularity and the positioning of the grid. A higher grid granularity results in higher runtime requirements but will most likely produce more accurate results.

SUBCLU [Kailing et al. 2004] uses the DBSCAN cluster model of density-connected sets [Ester et al. 1996]. It is shown that density-connected sets satisfy the downward closure property. This enables SUBCLU to search for density-based clusters in subspaces in an APRIORI-like style. The resulting clusters may exhibit an arbitrary shape and size in the corresponding subspaces. In fact,

for each subspace, SUBCLU computes all clusters that would have been found by DBSCAN applied to that subspace only. Compared to grid-based approaches, SUBCLU achieves a better clustering quality but requires a higher runtime.

It has been observed that a global density threshold, as used by SUBCLU and the grid-based approaches, leads to a bias towards a certain dimensionality: A tighter threshold which is able to separate clusters from noise well in low dimensions tends to lose clusters in higher dimensions, whereas a more relaxed threshold which is able to detect high-dimensional clusters will produce an excessive amount of low-dimensional clusters. Therefore, the dimensionality unbiased cluster model DUSC has been proposed, based on a density measure adaptive to the dimensionality [Assent et al. 2007a]. As a major drawback, this approach is lacking in antimonotonic properties and, thus, pruning the search space is not possible. A "weak density" is thus defined as a remedy, providing antimonotonic properties. This remedy, however, in turn introduces a global density threshold again. A method for visual subspace cluster analysis based on DUSC has been proposed by Assent et al. [2007b].

2.3.4 *Hybrid Clustering Algorithms.* Algorithms that do not aim at uniquely assigning each data point to a cluster nor at finding all clusters in all subspaces are called hybrid algorithms. Some hybrid algorithms offer the user an optional functionality of a pure projected clustering algorithm. Others aim at computing only the subspaces of potential interest rather than the final clusters. Usually, hybrid methods that report clusters allow overlapping clusters, but do not aim at computing all clusters in all subspaces.

DOC [Procopiuc et al. 2002] uses a global density threshold to define a subspace cluster by means of hypercubes of fixed side-length w containing at least α points. A random search algorithm is proposed to compute such subspace clusters from a starting seed of sampled points. A third parameter β specifies the balance between the number of points and the dimensionality of a cluster. This parameter affects the dimensionality of the resulting clusters, and thus DOC usually also has problems with subspace clusters of significantly different dimensionality. Due to the very simple clustering model, the clusters may contain additional noise points (if w is too large) or not all points that naturally belong to the cluster (if w is too small). One run of DOC may (with a certain probability) find one subspace cluster. If k clusters need to be identified, DOC has to be applied at least k times. If the points assigned to the clusters found so far are excluded from subsequent runs, DOC can be considered as a pure projected clustering algorithm because each point is uniquely assigned to one cluster or to noise (if not assigned to a cluster). On the other hand, if the cluster points are not excluded from subsequent runs, the resulting clusters of multiple runs may overlap. Usually, DOC cannot produce all clusters in all subspaces.

MINECLUS [Yiu and Mamoulis 2003, 2005] is based on a similar idea as DOC, but proposes a deterministic method to find an optimal projected cluster, given a sample seed point. The authors transform the problem into a frequent item set mining problem and employ a modified frequent pattern tree growth method. Further heuristics are introduced to enhance efficiency and accuracy.

DiSH [Achtert et al. 2007] follows a similar idea as PreDeCon but uses a hierarchical clustering model. This way, hierarchies of subspace clusters can be discovered, that is, the information that a lower-dimensional cluster is embedded within a higher-dimensional one. The distance between points and clusters reflects the dimensionality of the subspace that is spanned by combining the corresponding subspace of each cluster. As in COSA, the weighting of attributes is learned for each object, not for entire clusters. The learning of weights, however, is based on single attributes, not on the entire feature space. DiSH uses an algorithm that is inspired by the density-based hierarchical clustering algorithm OPTICS [Ankerst et al. 1999]. However, DiSH extends the cluster ordering computed by OPTICS in order to find hierarchies of subspace clusters with multiple inclusions (a lower-dimensional subspace cluster may be embedded in multiple higher-dimensional subspace clusters).

HARP [Yip et al. 2004] is a single-link-like hierarchical clustering algorithm but it uses a different "distance function" between points/clusters and does not produce a hierarchy of subspace clusters. Starting with singleton clusters, HARP iteratively merges clusters as long as the resulting cluster has a minimum number of relevant attributes. A relevance score is introduced for attributes based on a threshold that starts at some harsh value and is progressively decreased while clusters increase in size. By design, HARP has problems in finding low-dimensional clusters. The resulting dendrogram can be cut at any level in order to produce a unique assignment of points to clusters.

SCHISM [Sequeira and Zaki 2005] mines interesting subspaces rather than subspace clusters, hence, it is not exactly a subspace clustering algorithm, but solves a related problem: finding subspaces to look for clusters. It employs a grid-like discretization of the database and applies a depth-first search with backtracking to find maximally interesting subspaces.

FIRES [Kriegel et al. 2005] computes one-dimensional clusters using any clustering technique the user is most accomplished with in a first step. These one-dimensional clusters are then merged by applying a "clustering of clusters." The similarity of clusters is defined by the number of intersecting points. The resulting clusters represent hyper-rectangular approximations of the true subspace clusters. In an optional postprocessing step, these approximations can be refined by again applying any clustering algorithm to the points included in the approximation projected onto the corresponding subspace. Though using a bottom-up search strategy, FIRES is rather efficient because it does not employ a worst-case exhaustive search procedure but a heuristic that is linear in the dimensionality of the data space. However, this performance boost is paid for by an expected loss of clustering accuracy. It cannot be specified whether the subspace clusters produced by FIRES may overlap or not. In general, the clusters may overlap, but usually FIRES cannot produce all clusters in all subspaces.

P3C [Moise et al. 2008, 2006] starts with one-dimensional intervals that are likely to approximate higher-dimensional subspace clusters. These intervals are merged using an APRIORI-like bottom-up search strategy. The maximal-dimensional subspace cluster approximations resulting from this merging procedure are reported as so-called cluster cores. In a refinement step, the cluster cores are refined by using an EM-like clustering procedure. Each cluster core

is taken as one initial cluster for the EM algorithm. Points are assigned to the closest cluster core using the Mahalanobis distance. The final output of P3C is a matrix that records for each data point its probability of belonging to each projected cluster. From this matrix, a disjoint partitioning of the data points into clusters can be obtained by assigning each point to the cluster with the highest probability. If overlapping clusters shall be allowed, each point can be assigned to all clusters with a probability larger than $1/k$. P3C does not produce all clusters in all subspaces like any APRIORI-style algorithm does, but reports only the results of the final cluster computation step using EM.

Recently, Moise and Sander [2008] provided a first attempt to formulate the search for statistically significant subspace clusters as an optimization problem. In addition, the authors proposed an iterative algorithm called STATPC to search locally optimized solutions for this optimization problem.

2.4 Summary

Two schemata for the classification of algorithms for finding clusters in axis-parallel subspaces have been presented. The first schema classifies the approaches, according to the definition of the problem the algorithms aim to solve, into *projected clustering*, *subspace clustering*, and *hybrid algorithms*. The second schema distinguishes the algorithmic method to find the subspaces that accommodate the clusters, bottom-up versus top-down approaches. In fact, there is a close relationship between the problem-oriented classification and the algorithmic-oriented classification. Many projected clustering algorithms implement a top-down approach, whereas all subspace clustering algorithms follow a bottom-up approach. This close connection explains the additions "dimension-growth" and "dimension-reduction" in the distinction between "dimension-growth subspace clustering" and "dimension-reduction projected clustering" in the related chapter of Han and Kamber [2006]. However, this relationship does not hold in general. In addition, for hybrid approaches there is no close relationship to any of the algorithmic-oriented classes, that is, some implement a bottom-up approach, others use a top-down strategy.

A classification of existing approaches has been presented following the classification with respect to their assumed task definition. Table I overviews the different categorizations of algorithms and their relationships also for the algorithmic point of view. Note that DOC, MINECLUS, and P3C appear multiple times since they can optionally produce overlapping or nonoverlapping clusters. In addition, it is worth noting that some algorithms may also find nonaxis-parallel subspace clusters, such as P3C. This is usually a side-product of the implemented clustering model: P3C uses EM as a postprocessing step which also may find arbitrarily oriented subspace clusters. On the other hand, the clusters produced by approaches that use a cluster model accounting for clusters of arbitrary shape (i.e., density-based algorithms such as CLIQUE and SUBCLU) may have arbitrary (nonaxis-parallel or even nonlinear) shape in the relevant subspace but are still axis-parallel subspace clusters in the entire feature space (refer to Figure 5).

Table I. Categorization of Sample Subspace Clustering Algorithms
Projected Clustering Algorithms, Hybrid Approaches

		algorithmic-oriented view	
	category	**bottom-up**	**top-down**
problem-oriented view	**subspace clustering**	CLIQUE nCluster ENCLUS MAFIA SUBCLU	
	hybrid	DiSH FIRES P3C SCHISM	DOC MINECLUS COSA HARP
	projected clustering	P3C	PROCLUS SSPC PreDeCon DOC MINECLUS

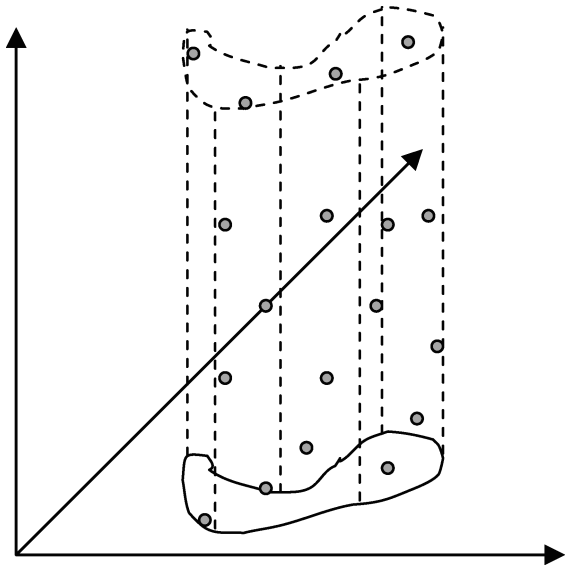


Fig. 5. Axis-parallel subspace cluster with arbitrary shape in the corresponding subspace.

In general, focusing on axis-parallel subspaces is meaningful in several applications. Since all existing approaches are based on further assumptions, a fair and comprehensive experimental comparison of all approaches is a large challenge. In most papers on axis-parallel subspace/projected clustering, the reader finds at most head-to-head comparisons between the newly proposed method and few other existing methods that often rely on completely different

assumptions. Since very often these differences are not discussed carefully, these comparisons are misleading. Parsons et al. [2004] compare very few methods for axis-parallel and (without drawing a distinction between the different tasks) also arbitrarily oriented subspace clustering experimentally. Although this is a first attempt in the right direction, this comparison is far from complete. However, to decide which algorithm should be chosen for which task, such a comparison is urgently needed. Leastwise, this survey is an attempt to provide such a comparison from the theoretical point of view, sketching the different assumptions and heuristics used by the various approaches.

3. FINDING CLUSTERS BASED ON PATTERNS IN THE DATA MATRIX

Recall the cluster definition of subspace and projected clustering algorithms: A clustering can be described as a set of pairs (X, Y), where X is a subset of data objects and Y is a subset of data attributes, such that the points in X are close when projected onto the attributes in Y, but projected onto the remaining attributes they are not close. Since the measure of closeness is unspecified by the problem definition of subspace and projected clustering in principle, also most pattern-based clustering algorithms could be interpreted as subspace or projected clustering algorithms in the aforesaid sense. The clustering algorithms surveyed in Section 2 usually define the "closeness" based on density in terms of the Euclidean distance in an axis-parallel projection. The pattern-based clustering algorithms, as we will see in this section, define the closeness differently in terms of a common behavior of objects in an axis-parallel subspace, that is, with respect to a certain "pattern" which the objects form in a subset of attributes.

Let us embark upon discussing pattern-based clustering algorithms with a general consideration. We have seen that the heuristics used in subspace and projected clustering treat dimensions and points differently. Why are those directions not interchangeable? A reason may be that heuristics for speedup are based on different intuitions for data space and data objects. Furthermore, depending on the problem at hand, the spatial intuition may be natural. Thus, data space and data objects are indeed different concepts for many applications. However, it is a general characteristic of pattern-based clustering algorithms (thus they are also called biclustering, coclustering, two-mode clustering, or two-way clustering algorithms) that they treat attributes and objects interchangeably. Note that there are counter-examples, though. The algorithm MaPle [Pei et al. 2003] (see the following) enumerates attributes first, based on the reasoning that there are usually much more objects than attributes in a database.

While we claim to provide a rather thorough overview of existing approaches in Sections 2 and 4, in this section we aim merely at pointing out the connections among different biclustering models and their relationships to the more general approaches based on spatial intuitions. To present an overview on different models and to connect these models to spatial intuitions, we follow the structure presented by Madeira and Oliveira [2004] and try to enrich the rather abstract notions of bicluster types, by intuitions about what the patterns in a

data matrix mean in the original data space. This will lead us to the surprising perception that, in terms of general subspace clustering approaches, many approaches in this field tackle rather simple, very specialized, or even weird problems. For a more exhaustive covering of biclustering algorithms we refer to the aforementioned surveys covering biclustering in biological and medical applications [Madeira and Oliveira 2004; Van Mechelen et al. 2004; Tanay et al. 2006]. Recent work on pattern-based clustering is especially popular in the bioinformatics community, focusing on the application of biclustering on microarray data, triggered by Cheng and Church [2000].

3.1 General Aim and Basic Approaches of Pattern-Based Clustering Algorithms

Pattern-based clustering algorithms depict the data as a matrix $\boldsymbol{A}$ with a set of rows X and a set of columns Y. The element a_{xy} represents the value in row x and column y. Usually, the rows represent database objects, and the columns are the attributes of the database objects. Thus the matrix element a_{xy} is the value of object with ID x in the attribute with ID y. We can consider such a matrix $\boldsymbol{A}$, with n rows and m columns, defined by its set of rows, $X = \{x_1, \ldots, x_n\}$, and its set of columns, $Y = \{y_1, \ldots, y_m\}$. Thus we can denote the matrix $\boldsymbol{A}$ by (X, Y). Choosing $I \subseteq X$ and $J \subseteq Y$ as subsets of the rows and columns, respectively, $\boldsymbol{A}_{IJ} = (I, J)$ denotes the submatrix of $\boldsymbol{A}$ containing those elements a_{ij} with $i \in I$ and $j \in J$. Biclustering algorithms tackle the problem of finding a set of submatrices $\{(I_1, J_1), \ldots, (I_k, J_k)\}$ of the matrix $\boldsymbol{A} = (X, Y)$ (with $I_i \subseteq X$, $J_i \subseteq Y$ $\forall i \in \{1, \ldots, k\}$), where each submatrix (bicluster) meets a given homogeneity criterion.

Many approaches make use of mean values of rows, of columns, and of the complete data matrix or a certain submatrix (i.e., a bicluster). For these mean values, the following notations are commonly in use. The mean of the ith row in the bicluster (I, J) is given by

$$a_{iJ} = \frac{1}{|J|} \sum_{j \in J} a_{ij}. \tag{1}$$

The mean of the jth column in the bicluster (I, J) is given by

$$a_{Ij} = \frac{1}{|I|} \sum_{i \in I} a_{ij}. \tag{2}$$

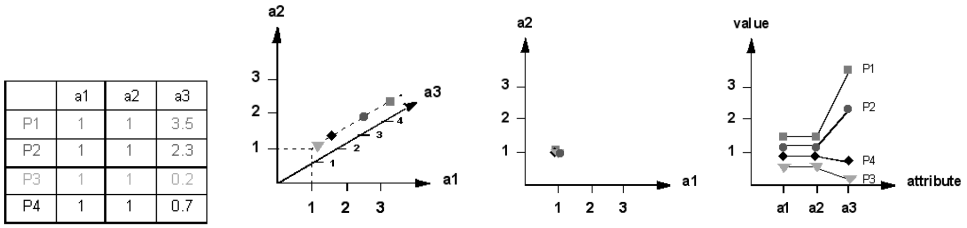
The mean of all elements in the bicluster (I, J) is given by

$$a_{IJ} = \frac{1}{|I||J|} \sum_{i \in I, j \in J} a_{ij} \tag{3}$$

$$= \frac{1}{|I|} \sum_{i \in I} a_{iJ} \tag{4}$$

$$= \frac{1}{|J|} \sum_{j \in J} a_{Ij}. \tag{5}$$

Madeira and Oliveira [2004] basically discern four different categories of biclusters: *constant biclusters*, *biclusters with constant values on either columns*



	a1	a2	a3
P1	1	1	3.5
P2	1	1	2.3
P3	1	1	0.2
P4	1	1	0.7

(a) data matrix $\boldsymbol{A_{XY}}$ (b) data space (c) subspace $\{a_1, a_2\}$ (d) pattern

Fig. 6. Constant bicluster.

or rows, *biclusters with coherent values*, and finally *biclusters with coherent evolutions*. The general problem settings for these categories are discussed in the following.

3.1.1 *Constant Biclusters.* A *perfect* constant bicluster consists of points sharing identical values in all selected attributes. In the corresponding submatrix (I, J) it holds therefore for a constant value μ which is typical for the cluster and for all $i \in I$ and $j \in J$:

$$a_{ij} = \mu. \tag{6}$$

In a not-so-perfect constant bicluster the values are only similar but not necessarily identical, that is,

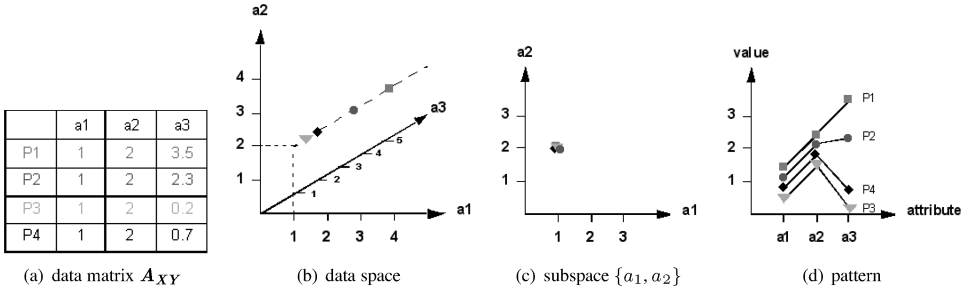
$$a_{ij} \approx \mu. \tag{7}$$

This type of bicluster is obviously an axis-parallel subspace cluster. Projecting the contributing points onto the contributing attributes, the points cluster at one single point. However, this single point is a special case, since it has identical attribute values in all directions; hence, it is always located on the bisecting line of the subspace relevant to the cluster (see Figure 6).

For the following categories, we will focus on perfect biclusters. Generally, however, real-world biclusters cannot be expected to be perfect; the model will rather apply only approximately on the data. Allowing for imprecise models makes the task of finding biclusters even harder. We have to decide when a cluster satisfactorily suffices the model in its general form. For example, optimizing for perfect constant biclusters on the matrix $\boldsymbol{A} = (X, Y)$ will probably lead to $|X| \cdot |Y|$ biclusters, each consisting of only one point and one dimension. How to avoid this kind of overfitting constitutes one interesting question of different contributions to this task.

3.1.2 *Biclusters with Constant Values on Rows or Columns*

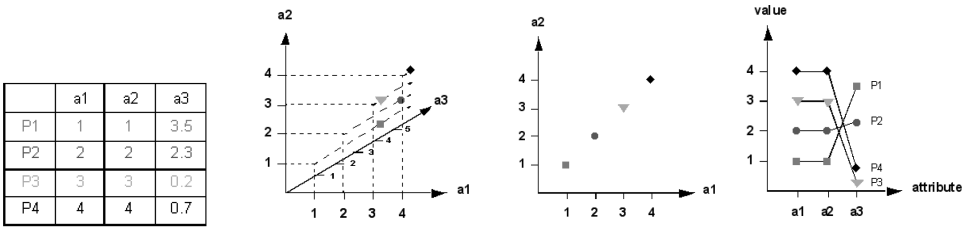
—Biclusters with Constant Values on Columns. Biclusters of points sharing constant values on columns are a more relaxed case of axis-parallel subspace clusters. The projection onto the contributing attributes yields once again a single point, but this point can be arbitrarily located anywhere in the corresponding subspace (see Figure 7). For the corresponding submatrix $\boldsymbol{A}_{IJ} = (I, J)$ it holds for a constant value μ which is typical for the cluster, with an adjustment



	a1	a2	a3
P1	1	2	3.5
P2	1	2	2.3
P3	1	2	0.2
P4	1	2	0.7

(a) data matrix $\boldsymbol{A_{XY}}$ (b) data space (c) subspace $\{a_1, a_2\}$ (d) pattern

Fig. 7. Constant values on columns.



	a1	a2	a3
P1	1	1	3.5
P2	2	2	2.3
P3	3	3	0.2
P4	4	4	0.7

(a) data matrix $\boldsymbol{A_{XY}}$ (b) data space (c) subspace $\{a_1, a_2\}$ (d) pattern

Fig. 8. Constant values on rows.

value c_j for column $j \in J$, and for all $i \in I$ and $j \in J$:

$$a_{ij} = \mu + c_j. \tag{8}$$

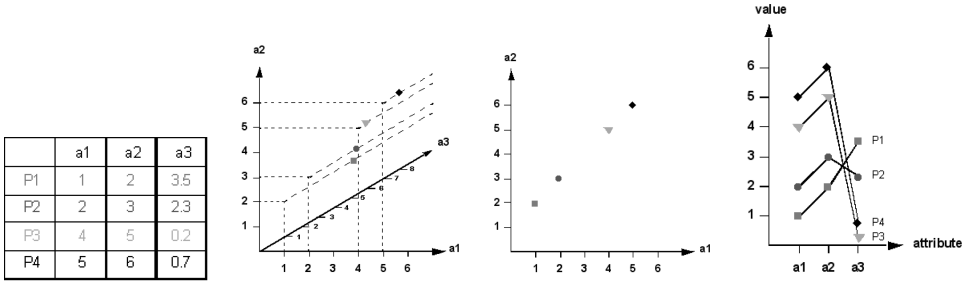
—*Biclusters with Constant Values on Rows.* Biclusters with constant values on rows accommodate the participating points on the bisecting line of the participating dimensions (refer to Figure 8). For the corresponding submatrix $\boldsymbol{A}_{IJ} = (I, J)$ it holds for a constant value μ which is typical for the cluster, with an adjustment value r_i for row $i \in I$, and for all $i \in I$ and $j \in J$:

$$a_{ij} = \mu + r_i. \tag{9}$$

3.1.3 *Biclusters with Coherent Values.* More sophisticated approaches seek biclusters with coherent values exhibiting a particular form of covariance between rows and columns. One way to describe such biclusters is as a combination of Eqs. (8) and (9). For a perfect bicluster with coherent values, (I, J), the values a_{ij} can be predicted by an additive model as

$$a_{ij} = \mu + r_i + c_j. \tag{10}$$

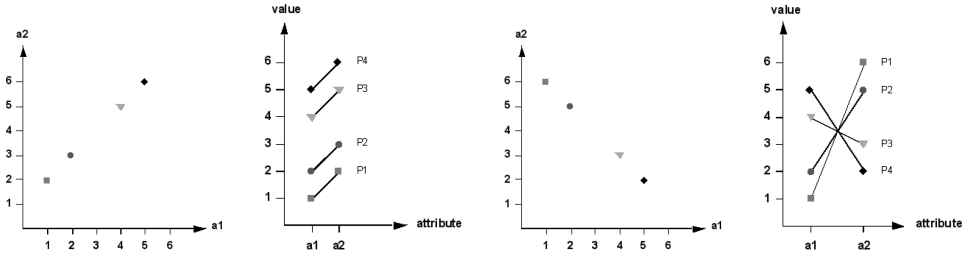
Again, r_i is an adjustment value for row $i \in I$ and c_j is an adjustment value for column $j \in J$. The difference from the simpler model of constant values in rows or columns is constituted by using both adjustment values simultaneously to adjust the mean value μ to a certain value in row i and column j. In fact, biclusters with constant values in columns or rows, respectively, could be regarded as special cases of biclusters with coherent values, where the adjustment values are $r_i = 0$ (resulting in Eq. (8)) or $c_j = 0$ (resulting in Eq. (9)).



	a1	a2	a3
P1	1	2	3.5
P2	2	3	2.3
P3	4	5	0.2
P4	5	6	0.7

(a) data matrix A_{XY} (b) data space (c) subspace $\{a_1, a_2\}$ (d) pattern

Fig. 9. Coherent values.



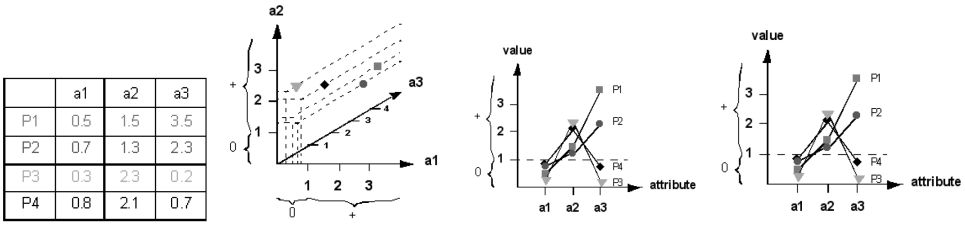
(a) positive correlation (b) pattern (c) negative correlation (d) pattern

Fig. 10. Patterns corresponding to positively and negatively correlated attributes.

The corresponding clusters accommodate data points on hyperplanes parallel to the axes of irrelevant attributes in the complete data space. Projected onto the corresponding subspace, the clusters appear as increasing one-dimensional lines (see Figure 9). This pattern includes constant lines, which reduces to the special case of the category of biclusters with constant values on columns.

Note that decreasing lines are not covered by this model because these would result in a completely different pattern in the data matrix: While increasing lines consist of positively correlated attributes, decreasing lines result from negatively correlated attributes. The corresponding pattern cannot be described by the simple additive models typical for biclustering approaches, which only cover shifted patterns (refer to Figure 10).

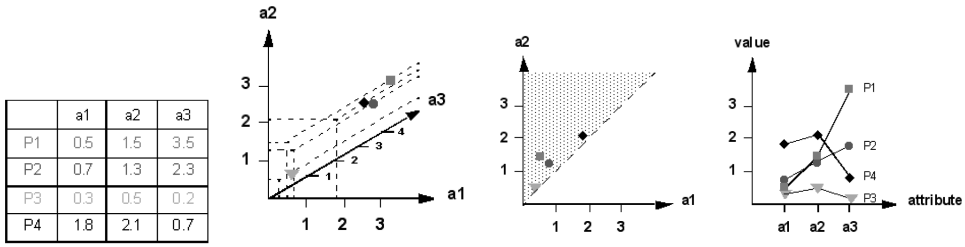
3.1.4 *Biclusters with Coherent Evolutions.* In this category, biclusters are constituted by a set of rows and columns, where the changes of attribute values are common among attribute pairs for all participating rows not in the exact quantity, but only in the fact that a change happens at all. Some approaches require the change of attribute values to exhibit the same direction (either increasing or decreasing). Some approaches quantize the occurring attribute values in some discrete states and address equal state-transitions (e.g., Murali and Kasif [2003], Tanay et al. [2002]). To obtain an intuition behind such bicluster models, imagine a quantizing approach with some states. The set of states constitutes a grid in the data space. A bicluster then contains a set of



	a1	a2	a3
P1	0.5	1.5	3.5
P2	0.7	1.3	2.3
P3	0.3	2.3	0.2
P4	0.8	2.1	0.7

(a) data matrix $\boldsymbol{A_{XY}}$ (b) data space (c) subspace $\{a_1, a_2\}$ (d) pattern

Fig. 11. Coherent evolutions: state transitions.



	a1	a2	a3
P1	0.5	1.5	3.5
P2	0.7	1.3	2.3
P3	0.3	0.5	0.2
P4	1.8	2.1	0.7

(a) data matrix $\boldsymbol{A_{XY}}$ (b) data space (c) subspace $\{a_1, a_2\}$ (d) pattern

Fig. 12. Coherent evolutions: change in the same direction.

points I that fill the same grid cell in the projection of the set of attributes J contributing to the bicluster (see Figure 11).

Since quantizing approaches could bluntly be regarded as grid-based, axis-parallel subspace clustering (see Figure 11(c)); we will rather inspect an approach seeking clusters exhibiting a general tendency in attribute values as an example for biclustering with coherent evolution patterns. This phenomenon has been grasped as the *order-preserving submatrix* problem [Ben-Dor et al. 2002]. The idea is to find a subset of rows and columns where a permutation of the set of columns exists such that the values in every row are increasing.

For this model, we find no spatial intuition corresponding to the visualizations given so far (see Figure 12). However, since all points are located in a half-space of the relevant subspace (refer to Figure 12(c)), we may probably find related approaches in the field of quantitative association rule mining (refer to Webb [2001], Rückert et al. [2004], Georgii et al. [2005]) or in the adaptation of formal concept analysis [Ganter and Wille 1999] to numeric data [Pfaltz 2007].

3.2 Pattern-Based Clustering Algorithms

3.2.1 *Constant Biclusters.* Hartigan [1972] provided the classical description of the biclustering problem. The quality of a bicluster is given by the sum of squares of all entries, assuming the average value to form the corresponding ideal (perfect) bicluster. This could also be regarded as the variance of the submatrix $\boldsymbol{A}_{IJ}$, given the mean a_{IJ} (see Eq. (3)):

$$VAR(\boldsymbol{A}_{IJ}) = \sum_{i \in I, j \in J} (a_{ij} - a_{IJ})^2. \tag{11}$$

The data matrix is split recursively into two partitions. At each step, the split is chosen maximizing the reduction in the overall sum of squares of all biclusters. The splitting stops when the reduction of the sum of squares is less than that expected by chance.

This procedure is similar to a divisive, top-down hierarchical clustering, and therefore results in a rather inefficient procedure.

3.2.2 *Biclusters with Constant Values in Rows or Columns.* Algorithms of this category usually apply a normalization to transform the biclusters into constant biclusters (e.g., Getz et al. [2000]). Other approaches described in the bioinformatics community consider the existence of multiplicative noise, or constrain the values in rows and columns to certain intervals, or even provide probabilistic models for the clusters (e.g., Califano et al. [2000], Sheng et al. [2003], Segal et al. [2001]).

Besides these application-driven methods, we know of no general biclustering approach to this specific problem. However, the problem seems not that intriguing after all, since it can easily be reduced to the first category, and, in turn, the following category treats problems belonging to this category as special cases.

Let us note that the general subspace and projected clustering algorithms described in Section 2 tackle the problem of finding biclusters with constant values on columns in a general way. The problem of finding biclusters with constant values on rows is a very special case of general correlation clustering algorithms (refer to Section 4).

3.2.3 *Biclusters with Coherent Values.* Cheng and Church [2000] are credited with having introduced the term *biclustering* (inspired by [Mirkin 1996]) to the analysis of gene expression (microarray) data. They assess the quality of a bicluster (I, J) by a *mean squared residue* value H, given by

$$H(I, J) = \frac{1}{|I| \cdot |J|} \sum_{i \in I, j \in J} (a_{ij} - a_{iJ} - a_{Ij} + a_{IJ})^2. \tag{12}$$

The submatrix (I, J) is then considered a δ-bicluster if $H(I, J) \leq \delta$ for a given $\delta \in \mathbb{R}_0^+$. Setting $\delta = 0$ results in perfect δ-biclusters. In this model, a bicluster is perfect if each row and column exhibits an absolutely consistent bias. The bias of column j with respect to the other columns is given by $a_{Ij} - a_{IJ}$. The bias of row i with respect to the other rows is given by $a_{iJ} - a_{IJ}$. In a perfect δ-bicluster, the value a_{ij} is then given additively by a row-constant, a column-constant, and an overall constant value.

$$a_{ij} = a_{iJ} + a_{Ij} - a_{IJ} \tag{13}$$

Setting $\mu = a_{IJ}$, $r_i = a_{iJ} - a_{IJ}$, and $c_j = a_{Ij} - a_{IJ}$, this model corresponds to the general description of additive models for biclusters with coherent values given by Eq. (10). However, the value a_{ij} is not directly given as in Eq. (13) whenever a δ-bicluster is *not* perfect. In this case, the value predicted by the model will deviate from the true model.

$$a_{ij} = res(a_{ij}) + a_{iJ} + a_{Ij} - a_{IJ} \tag{14}$$

Equivalently, the residue is given by

$$res(a_{ij}) = a_{ij} - a_{iJ} - a_{Ij} + a_{IJ}. \tag{15}$$

This value is used in Eq. (12) to calculate the mean squared residue. In a similar way, the mean squared residue of a row i or of a column j, respectively, is defined as

$$d(i) = \frac{1}{|J|} \sum_{j \in J} (a_{ij} - a_{iJ} - a_{Ij} + a_{IJ})^2, \tag{16}$$

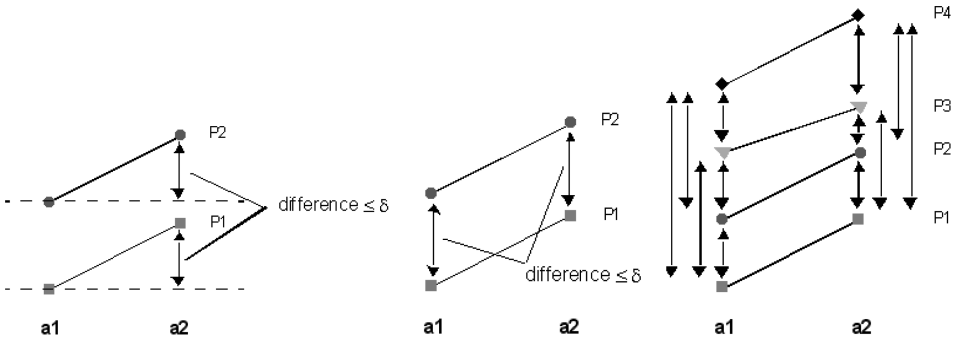
$$d(j) = \frac{1}{|I|} \sum_{i \in I} (a_{ij} - a_{iJ} - a_{Ij} + a_{IJ})^2. \tag{17}$$

To find a δ-bicluster, Cheng and Church [2000] propose to greedily remove the row or column (or a set of rows or columns) with maximal mean squared residue of the row or the column, until the remaining submatrix (I, J) satisfies $H(I, J) \leq \delta$. Afterwards, in order to find a *maximal* δ-bicluster, rows and columns are added to (I, J) unless adding the row or column would increase the value $H(I, J)$. Curiously, if the number of rows or columns is below 100 (which is still a pretty high number of objects or a high-dimensionality, respectively, in many applications), no multiple row or column deletion is performed.

So far, the procedure finds one δ-bicluster in a data matrix. In order to find k δ-biclusters, the procedure is iterated k times. Any δ-bicluster already found has to be masked by random numbers. This makes it unlikely that elements covered by existing biclusters would contribute to any future bicluster, but does not remove complete rows and columns. The finally resulting set of k biclusters should therefore be disjunct with respect to combinations of rows and columns; that is, a data point cannot contribute to different clusters based on the same attributes, and an attribute cannot contribute to different clusters for the same data point. Thus, in our sense, the clusters may overlap and the subspaces may overlap, but, in theory, not both at the same time. Besides inserting random numbers, the algorithm is deterministic and should retrieve equal results in different runs. If not, the random numbers contributed to a cluster, which would be an alarming effect.

Several later contributions to the field are based on the δ-bicluster model as proposed by Cheng and Church [2000] but address some issues bequeathed by their approach. As weak points in the approach of Cheng and Church, we could state the following.

(1) One cluster at a time is found, then the cluster needs to be masked in order to find a second cluster.
(2) This procedure bears an inefficient performance.
(3) The masking may lead to less accurate results.
(4) The masking inhibits simultaneous overlapping of rows and columns.
(5) Missing values cannot be dealt with.
(6) The user must specify the number of clusters beforehand.



(a) visualization of inequality (18) (b) visualization of inequality (19) (c) pairwise differences uncover outliers

Fig. 13. P-cluster model: pairwise differences.

In some cases, however, whether a certain point is a "weak" point is a matter of taste. The stated weak points are addressed as such in several publications reported next.

With FLOC, Yang et al. [2002] introduce another algorithm to find δ-biclusters. FLOC is a randomized move-based algorithm efficiently approximating k δ-clusters, again based on the minimization of the average residue. Initial seed-clusters are optimized by randomly chosen steps of removing or adding a row or a column. This addresses issue 1. Furthermore, the algorithm allows simultaneous overlapping of rows and columns (issue 4). The model is adapted in taking only specified values into account for the computation of the residue, thus allowing for missing values, addressing issue 5. Another example of algorithms allowing for missing values has been proposed by Sim et al. [2006].

However, the improvements stated by Yang et al. [2002] are paid for by introducing random events. Therefore, the same authors propose also a deterministic approach with the p-cluster model [Wang et al. 2002]. This model specializes the δ-bicluster-property to a pairwise property of two objects on two attributes as

$$|(a_{i_1 j_1} - a_{i_1 j_2}) - (a_{i_2 j_1} - a_{i_2 j_2})| \leq \delta \tag{18}$$

or equivalently

$$|(a_{i_1 j_1} - a_{i_2 j_1}) - (a_{i_1 j_2} - a_{i_2 j_2})| \leq \delta. \tag{19}$$

Inequality (18) describes the difference between two objects by their relative differences of two attribute values (refer to Figure 13(a)). Inequality (19) describes the difference of two attributes by the absolute differences between two objects. Both conditions cover identical sets of cases. A submatrix (I, J) is a δ-p-cluster if this property is fulfilled for any 2×2-submatrix $(\{i_1, i_2\}, \{j_1, j_2\})$, where $\{i_1, i_2\} \subseteq I$ and $\{j_1, j_2\} \subseteq J$. Formulating the pattern description as a pairwise condition tightens the model for biclusters. While limiting the overall variance of a bicluster may allow to include some outliers in a cluster, the pairwise condition excludes outliers more rigorously (see Figure 13(c)).

After creating the maximal set of attributes for each pair of objects forming a δ-p-cluster and the maximal set of objects for each pair of attributes forming a

δ-p-cluster, a pruning-step is implemented to lower the impact of the final step, namely the search in the set of submatrices. This search, however, requires exponential time after all. Addressed issues with respect to the Cheng and Church approach are 1, 4, and 6.

Another related approach is MaPle [Pei et al. 2003], stressing the maximality of the mined δ-p-clusters, based on the closure property of subclusters coming along with the model of δ-p-clusters: For a δ-p-cluster (I, J), every submatrix (I', J') with $I' \subseteq I$ and $J' \subseteq J$ is a δ-p-cluster. Note that this is not necessarily true for the more general bicluster models of Cheng and Church [2000] or Yang et al. [2002], since an outlier may be covered by a bigger cluster but would influence the variance of a smaller cluster considerably (see Figure 13(c)). MaPle mines for maximal δ-p-clusters, that is, for a given δ-p-cluster (I, J) there exists no other δ-p-cluster (M, N) in the dataset with $I \subset M$ and $J \subset N$. Essentially, MaPle performs the analysis in a similar way as the previous approach [Wang et al. 2002], but uses the closure property for pruning any superset D' of the attribute set D once D is found unsuitable to serve as base for a p-cluster. Still, like the approach by Wang et al. [2002], also MaPle is based on a complete enumeration in the end. Thus, once again, the addressed issues of the Cheng and Church approach include items 1, 4, and 6.

The CoClus algorithms proposed by Cho et al. [2004] seek a marriage of a k-means-like approach with the cluster models of Hartigan or Cheng and Church. To avoid poor local minima and empty clusters, a local search strategy is implemented swapping single rows between clusters if this reduces the objective function. The addressed issues with respect to the Cheng and Church approach are therefore items 1 and 2. However, these algorithms cannot avoid the typical flaws of k-means-like approaches as being caught in local minima (despite the local search strategy), requiring specification of the number k of clusters beforehand, and, as a complete partition of the dataset onto k clusters that are disjunct with respect to rows as well as columns, the dataset is assumed to contain no noise. Instead, every attribute is assumed relevant for exactly one cluster. This assumption generally contradicts the circumstances of clustering high-dimensional data and the presence of noise will generally deteriorate the quality of the clustering result.

3.2.4 *Biclusters with Coherent Evolutions.* The concept of an Order Preserving Submatrix (OPSM), introduced by Ben-Dor et al. [2002], describes a submatrix (I, J) of the data matrix $\boldsymbol{A}$ where a permutation π of the set of columns J exists such that for each row $i \in I$ and each index $1 \leq m < |J|$ within the permutation $\pi(J)$ the following inequality holds:

$$a_{i\pi(J)[m]} < a_{i\pi(J)[m+1]}, \tag{20}$$

that is, according to the given linear order of columns, the values in the selected columns are strictly increasing. The cluster model is then given by the pair (J, π). The *support* of a model is the set I of rows fitting to the model according to inequality (20).

The algorithm of Ben-Dor et al. searches the best model in a greedy bottom-up-approach (i.e., starting with small models and iteratively extending the best

l of these models). The "best" model is the one with largest statistical significance (i.e., having the smallest prior probability). This algorithm favors models with a large support.

Liu and Wang [2003] follow the same general idea defining a bicluster as OP-Cluster (Order Preserving Cluster) but weaken the conditions of Ben-Dor et al. by introducing groups of similar attributes. They also discard the assessment of statistical significance and instead report all (maximal) submatrices covering at least a given minimum number of rows and columns. The algorithm creates a nondecreasing order of columns for each row (grouping together similar columns). The resulting set of column-sequences is mined for frequent patterns.

As said previously, we do not have any spatial intuition explaining this model. The resulting biclusters consist of objects showing a similar trend on a set of attributes. Whether the corresponding attributes are correlated or even linearly correlated remains unclear. The results may be interesting in some application domains, but the clusters are not necessarily accommodated in any specific subspace of the data space. Considering the model as sketched earlier (see Inequality (20)), clearly there is no way to predict an attribute value for a given instance and a specified column. The model merely allows to predict the value to exceed a given threshold, namely the attribute value in the preceding column. Hence, the points occupy half-spaces and, as stated before, we find this approach being related to quantitative association rule mining.

3.3 Summary

Independent of the concrete problem formulation and the spatial intuition behind it, the biclustering problem is sometimes formulated as a graph mining problem (e.g., Dhillon [2001], Tanay et al. [2002], Sim et al. [2006]). The data matrix is then described as a bipartite graph, where one set of vertices corresponds to the rows, the other set corresponding to the columns. As an example problem formulation, finding a minimum set of biclusters to cover all the elements in a data matrix is a generalization of the problem of covering a bipartite graph by a minimum set of bicliques, a problem known to be NP-hard [Garey and Johnson 1979]. The exact complexity of a biclustering problem depends on the exact problem definition and the merit function used to evaluate the quality of a specific biclustering. For most of the common biclustering problems the computational complexity is not known. If, however, the computational complexity of a specific biclustering problem-formulation is known, it usually is an NP-hard problem. Thus, different heuristics, simplifying models, and greedy or randomized approaches are implemented. Let us note that, of course, clustering in general is also an NP-hard problem although we are used to efficient solutions. Every efficient clustering algorithm can thus only provide an approximative solution based on certain assumptions and heuristics, such as those reflected by its underlying cluster model.

Although biclustering models do not fit exactly into the spatial intuition behind subspace, projected, or correlation clustering (a summarizing comparison of patterns in the data matrix, corresponding bicluster models, and related

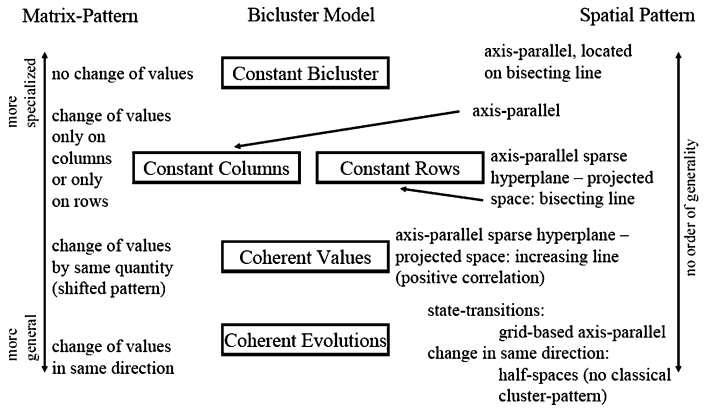


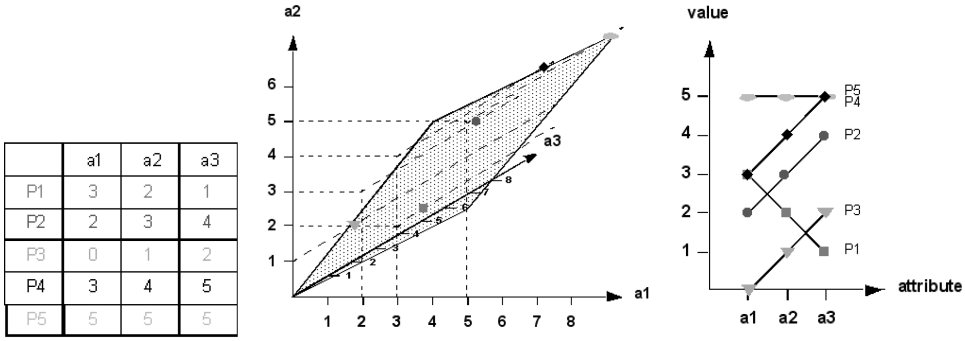
Fig. 14. Comparison: patterns in biclustering approaches and their corresponding spatial intuitions.

spatial patterns is given in Figure 14, these models make sense in view of a data matrix and fruitful applications seem to justify the approach. However, since the cluster models forming the basis of biclustering algorithms differ considerably, conducting a fair comparison among these algorithms is a nontrivial task. A thorough evaluation (let alone in comparison with axis-parallel and correlation clustering algorithms) is not available in the literature. Recently, Prelić et al. [2006] performed an evaluation of five selected methods [Cheng and Church 2000; Tanay et al. 2002; Ben-Dor et al. 2002; Ihmels et al. 2004; Murali and Kasif 2003]. However, a thorough comparison of clustering algorithms requires quite some effort since there is not a solely and uniquely established comparison method of clusterings. In view of the rather specialized tasks performed by biclustering approaches, the comparison of performance with respect to effectiveness as well as efficiency should be performed in a broad context of subspace clustering, projected clustering, and correlation clustering. Despite (or by virtue of) the specialization of the biclustering cluster models, this family of clustering approaches seems to perform well on microarray data. In this application domain, there exists a vast amount of approaches not covered in this survey (refer to several surveys focused on biological and medical applications [Van Mechelen et al. 2004; Jiang et al. 2004; Madeira and Oliveira 2004; Tanay et al. 2006]).

4. FINDING CLUSTERS IN ARBITRARILY ORIENTED SUBSPACES

4.1 General Aim of Algorithms in this Category

While the aim of pattern-based clustering algorithms is easily understood in terms of corresponding matrix representations, the spatial intuition behind these approaches is not quite so convincing. A clear pattern in a matrix corresponds to special, or even rather artificial, constellations of the corresponding points in space (or even to no specific constellation at all). The explaining models remain rather simple and cannot include simple negative correlations, let alone more complex correlations.



	a1	a2	a3
P1	3	2	1
P2	2	3	4
P3	0	1	2
P4	3	4	5
P5	5	5	5

(a) data matrix A_{XY} (b) data space (c) pattern

Fig. 15. Points exhibiting a linear correlation pattern among three attributes according to the linear equation $a_1 - 2 \cdot a_2 + a_3 = 0$.

A more general, intuitive approach is adopted by a family of algorithms known as *oriented clustering* or *generalized subspace/projected clustering* or *correlation clustering*[1] algorithms. These algorithms assume any cluster being located in an arbitrarily oriented subspace S of the data space $\mathbb{R}^d$. Such clusters appear as hyperplanes of arbitrary dimensionality in the data space. However, no typical pattern in the data matrix does correspond to these models based on a spatial intuition. For simple negative correlations, this has already been shown along with Figure 10. But compared to the relatively simple patterns describing positive correlations among pairs of attributes tackled by biclustering approaches for coherent values (refer to Section 3.1.3), complex positive and negative correlations among several attributes, resulting in hyperplanes in the data space, are even less regular when described as patterns as used for pattern-based approaches (see Figure 15). Thus, the approaches based on the intuitions typical for biclustering algorithms are in principle not suitable to tackle this more general problem, albeit some bicluster models can be regarded as special cases of correlation clustering models.

In terms of subspace clustering, an affine subspace $S + \vec{a}$, $S \subset \mathbb{R}^d$ with affinity $\vec{a} \in \mathbb{R}^d$ is interesting if a set of points exhibits a certain density within this subspace (i.e., projected onto this subspace), regardless of high variances along axes in the perpendicular subspace $(\mathbb{R}^d \setminus S) + \vec{a}$. In *oriented clustering*, these interesting subspaces need not be axis-parallel, but can be arbitrarily oriented (see Figure 16).

Assuming high variance along those axes forming the perpendicular subspace $(\mathbb{R}^d \setminus S) + \vec{a}$, the points building the cluster in S fill the perpendicular subspace $(\mathbb{R}^d \setminus S) + \vec{a}$) to a certain extension. In the complete data space $\mathbb{R}^d$, the points therefore form a hyperplane which is located in the subspace $(\mathbb{R}^d \setminus S) + \vec{a}$ (see Figure 17).

[1]Note that the term "correlation clustering" relates to a different task in the machine learning community, where a partitioning of the data shall correlate as much as possible with a pairwise similarity function f learned from past data (e.g., see Bansal et al. [2004]).

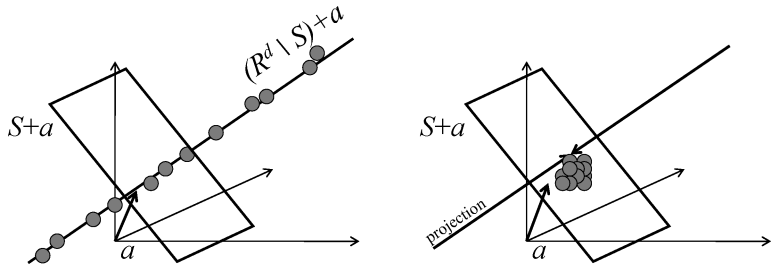


Fig. 16. Points distributed in data space where some attributes that are correlated cluster densely in a certain projection (arbitrarily oriented subspace).

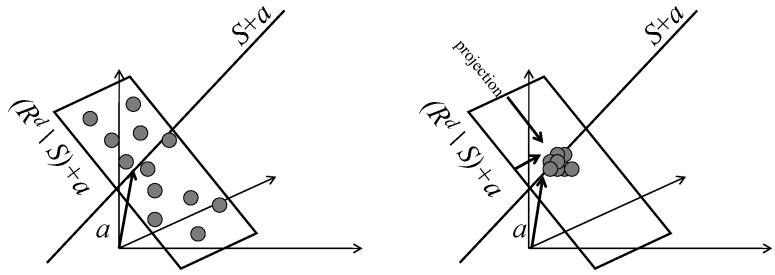


Fig. 17. Points distributed in an arbitrarily oriented subspace form a hyperplane.

Nevertheless, this observation facilitates an alternative way of describing *oriented clustering*. Points accommodated on a common hyperplane in data space appear to follow linear dependencies among the attributes participating in the description of the hyperplane. Since linear dependencies result in the observation of strong linear correlations among these attributes, we call this type of clustering also *correlation clustering*.

In fact, describing correlation clusters in terms of a subspace (hyperplane) accommodating the points rather than in terms of the subspace, where the points show high density, opens up different possibilities. Consider again the simple problem introduction in Figure 2 (Section 1). If some arbitrarily oriented subspaces are chosen to project the points and to search for dense sets of points, in this example, again all points would cluster densely in all three of the chosen subspaces. However, the points of, for example, cluster 1 and cluster 2 in Figure 2 are accommodated on different hyperplanes orthogonal to the chosen subspaces. This means that although the points would cluster densely in some lower-dimensional projections (the "interesting subspaces"), they do not belong to the same cluster if we take a different behavior in terms of complex correlations among subsets of attributes into account. Following a certain set of linear dependencies among attributes is the main characteristic of correlation clusters. Thus, considering the subspaces orthogonal to the "interesting subspace" of certain affinity (the so-called *correlation cluster hyperplanes*, i.e., arbitrarily oriented subspaces where a set of points exhibits high variance) enables to meaningfully discern different clusters that would have been merged if only the "interesting subspace" of high density would have been considered.

A common technique to grasp arbitrarily oriented directions of high variance in a dataset is Principal Component Analysis (PCA; for a thorough introduction see Jolliffe [2002]). Again, a local application of PCA is required in order to find clusters located in different subspaces (refer to the general problem statement in Section 1.2). Generally, applying PCA to a local selection of points is again based on the locality assumption. It is assumed that the hyperplane accommodating the points of a correlation cluster is sufficiently reflected in a local selection of points (e.g., the ε-neighborhood or the k-nearest neighbors of a point).

As a general idea, assume we apply a PCA-based approach to a given set of points $\mathcal{D} \subset \mathbb{R}^d$ in order to find the directions of high and low variance. First, we build the covariance matrix $\mathbf{\Sigma}_\mathcal{D}$ of $\mathcal{D}$. We have

$$\mathbf{\Sigma}_\mathcal{D} = \frac{1}{|\mathcal{D}|} \cdot \sum_{\vec{x} \in \mathcal{D}} (\vec{x} - \vec{x}_\mathcal{D}) \cdot (\vec{x} - \vec{x}_\mathcal{D})^\mathrm{T}, \tag{21}$$

where $\vec{x}_\mathcal{D}$ denotes the centroid (mean) of all points $\vec{x} \in \mathcal{D}$. $\mathbf{\Sigma}_\mathcal{D}$ is a $d \times d$ symmetric positive semidefinite matrix where $\sigma_{\mathcal{D}ij}$ (i.e., the value at row i and column j in $\mathbf{\Sigma}_\mathcal{D}$) equals the covariance between the dimensions i and j. The diagonal entry $\sigma_{\mathcal{D}ii}$ corresponds to the variance of the ith dimension. $\mathbf{\Sigma}_\mathcal{D}$ can be decomposed (by PCA) into the *eigenvalue* matrix $\boldsymbol{E}_\mathcal{D}$ of $\mathbf{\Sigma}_\mathcal{D}$ and the *eigenvector* matrix $\boldsymbol{V}_\mathcal{D}$ of $\mathbf{\Sigma}_\mathcal{D}$ such that

$$\mathbf{\Sigma}_\mathcal{D} = \boldsymbol{V}_\mathcal{D} \cdot \boldsymbol{E}_\mathcal{D} \cdot \boldsymbol{V}_\mathcal{D}^\mathrm{T}. \tag{22}$$

The eigenvalue matrix $\boldsymbol{E}_\mathcal{D}$ is a diagonal matrix holding the eigenvalues of $\mathbf{\Sigma}_\mathcal{D}$ in decreasing order in its diagonal elements. The eigenvector matrix $\boldsymbol{V}_\mathcal{D}$ is an orthonormal matrix with the eigenvectors of $\mathbf{\Sigma}_\mathcal{D}$ ordered correspondingly to the eigenvalues in $\boldsymbol{E}_\mathcal{D}$. The eigenvectors provide a new orthonormal basis. The eigenvalue matrix $\boldsymbol{E}_\mathcal{D}$ can be understood as the covariance matrix of the original dataset when represented in the new axis system $\boldsymbol{V}_\mathcal{D}$. All nondiagonal entries equal zero, meaning that all covariances have been removed. The first eigenvector in $\boldsymbol{V}_\mathcal{D}$ points to the direction of the highest variance in the dataset $\mathcal{D}$. The second eigenvector points to the direction of the second highest variance in $\mathcal{D}$ perpendicular to the first eigenvector. Assuming $\mathcal{D}$ being a correlation cluster, the major axes in $\boldsymbol{V}_\mathcal{D}$, say the first λ eigenvectors, span the λ-dimensional hyperplane accommodating the points of $\mathcal{D}$. We denote the first λ eigenvectors of $\boldsymbol{V}_\mathcal{D}$ by $\check{\boldsymbol{V}}_\mathcal{D}$ and call them *strong* eigenvectors. The remaining eigenvectors are called *weak* eigenvectors (denoted by $\hat{\boldsymbol{V}}_\mathcal{D}$). The weak eigenvectors can equivalently define the hyperplane accommodating the points of $\mathcal{D}$, as they all are orthogonal to that hyperplane. While the trace $\sum_{i=1}^{d} \sigma_{\mathcal{D}ii}$ is invariant under the axis transformation defined by the eigensystem $\boldsymbol{V}_\mathcal{D}$ (i.e., $\sum_{i=1}^{d} \sigma_{\mathcal{D}ii} = \sum_{i=1}^{d} e_{\mathcal{D}ii}$), the sum of the smallest $d - \lambda$ eigenvalues $\sum_{i=\lambda+1}^{d} e_{\mathcal{D}ii}$ is the minimum under all possible transformations. Thus, the smallest $d - \lambda$ eigenvectors define the subspace perpendicular to the hyperplane accommodating the cluster members, where the projected points would cluster optimally dense.

How to combine PCA (or related means to discern different subspaces) with the selection of interesting subspaces, with a suitable definition of λ, with the

selection of points, and with distance measures mainly makes the difference among the most common approaches to the task of correlation clustering.

What means soever are used to find correlation clusters, the general meaning of this family of clusters results in a mathematically clear, explanatorily rich, and predictively powerful model for correlation clusters [Achtert et al. 2006]: The λ-dimensional hyperplane accommodating the points of a correlation cluster $\mathcal{C} \subset \mathbb{R}^d$ can be defined by a linear equation system consisting of $d - \lambda$ equations for d variables, and the affinity, for example, given by the mean point $\vec{x}_\mathcal{C} = (\bar{x}_1 \cdots \bar{x}_d)^\mathrm{T}$ of all cluster members. We have

$$\begin{aligned}
v_{1(\lambda+1)} \cdot (x_1 - \bar{x}_1) + v_{2(\lambda+1)} \cdot (x_2 - \bar{x}_2) + \cdots + v_{d(\lambda+1)} \cdot (x_d - \bar{x}_d) &= 0 \\
v_{1(\lambda+2)} \cdot (x_1 - \bar{x}_1) + v_{2(\lambda+2)} \cdot (x_2 - \bar{x}_2) + \cdots + v_{d(\lambda+2)} \cdot (x_d - \bar{x}_d) &= 0 \\
&\vdots \\
v_{1d} \cdot (x_1 - \bar{x}_1) \quad + \quad v_{2d} \cdot (x_2 - \bar{x}_2) \quad + \cdots + \quad v_{dd} \cdot (x_d - \bar{x}_d) \quad &= 0
\end{aligned}$$

where v_{ij} is the value at row i, column j in the eigenvector matrix $\boldsymbol{V}_\mathcal{C}$ derived by PCA from the covariance matrix of $\mathcal{C}$. As introduced generally before, the first λ eigenvectors (i.e., the strong eigenvectors) give the directions of high variance and span the hyperplane accommodating $\mathcal{C}$. The remaining $d - \lambda$ weak eigenvectors span the perpendicular subspace. The linear equation system as sketched previously can therefore be given by

$$\hat{\boldsymbol{V}}_\mathcal{C}^\mathrm{T} \cdot \vec{x} \;=\; \hat{\boldsymbol{V}}_\mathcal{C}^\mathrm{T} \cdot \vec{x}_\mathcal{C}. \tag{23}$$

The defect of $\hat{\boldsymbol{V}}_\mathcal{C}^\mathrm{T}$ gives the number of free attributes, and the remaining attributes may actually be involved in linear dependencies. The equation system is by construction at least approximately fulfilled for all points $\vec{x} \in \mathcal{C}$ and provides a quantitative model for the cluster.

However, from the user's point of view of the application of correlation clustering on a new dataset, the correlations among attributes are observable, while the linear dependencies are merely an assumption to explain the correlations. Whether or not this assumption is valid can be evaluated by using the model as a predictive model and by refining the experiments based on the insights provided by the quantitative model (refer to Achtert et al. [2006]). Thus, this model of clustering is not only far more general than the models for biclustering sketched in Section 3, but also more concise and meaningful. An equivalently defined model has been apparently independently proposed within the pattern recognition community (refer to Haralick and Harpaz [2005], Harpaz [2007], Harpaz and Haralick [2007a]).

4.2 Correlation Clustering Algorithms

4.2.1 *PCA-Based Approaches.* The broad majority of correlation clustering approaches are based on an application of PCA on subsets of points (like range queries or k-nearest neighbor queries). The application of PCA largely dominates the time complexity of these algorithms, usually being cubic in the number of dimensions. The time complexity with respect to the number of data points differs due to different clustering schemes but usually is in $O(n)$ or at most $O(n^2)$.

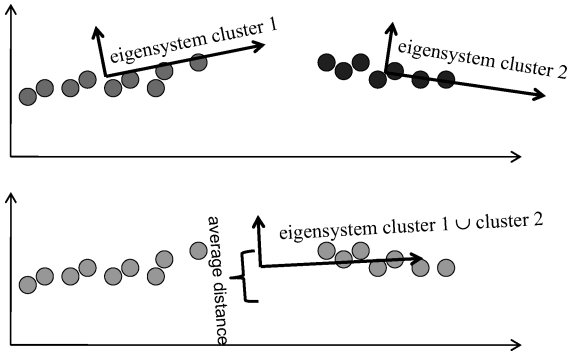


Fig. 18. ORCLUS: distance of two clusters.

As the first approach to *generalized projected clustering*, Aggarwal and Yu [2000] proposed the algorithm ORCLUS, using ideas similar to the axis-parallel approach PROCLUS [Aggarwal et al. 1999]. ORCLUS is a k-means-like approach, picking $k_c > k$ seeds at first, assigning the data base objects to these seeds according to a distance function that is based on an eigensystem of the corresponding cluster assessing the distance along the weak eigenvectors only (i.e., the distance in the projected subspace where the cluster objects exhibit high density). The eigensystem is iteratively adapted to the current state of the updated cluster. The number k_c of clusters is reduced iteratively by merging closest pairs of clusters until the user-specified number k is reached. The closest pair of clusters is the pair with the least average distance in the projected space (spanned by the weak eigenvectors) of the eigensystem of the merged clusters (see Figure 18). Starting with a higher k_c increases the effectiveness, but also the runtime. The method proposed by Chakrabarti and Mehrotra [2000] is a slight variant of ORCLUS designed for enhancing multidimensional indexing. Another, presumably more efficient, variant has been proposed by Li et al. [2007].

In contrast to ORCLUS, the algorithm 4C [Böhm et al. 2004a] is based on a density-based clustering paradigm [Ester et al. 1996]. Thus, the number of clusters is not decided beforehand, but clusters grow from a seed as long as a density criterion is fulfilled. Otherwise, another seed is picked to start a new cluster. The density criterion is a required minimal number of points within the neighborhood of a point, where the neighborhood is ascertained based on distance matrices computed from the eigensystems of two points. The eigensystem of a point $\vec{p}$ is based on the covariance matrix of the ε-neighborhood of $\vec{p}$ in Euclidean space. A parameter δ discerns large from small eigenvalues. In the eigenvalue matrix $\boldsymbol{E}_{\vec{p}}$ then large eigenvalues are replaced by 1, small eigenvalues by a value $\kappa \gg 1$. Using the adapted eigenvalue matrix $\boldsymbol{E'}_{\vec{p}}$, a correlation similarity matrix for $\vec{p}$ is obtained by $\boldsymbol{V}_{\vec{p}} \cdot \boldsymbol{E'}_{\vec{p}} \cdot \boldsymbol{V}_{\vec{p}}^{\mathrm{T}}$. This matrix is then used to derive the distance of two points, $\vec{q}$ and $\vec{p}$, with respect to $\vec{p}$, as the general quadratic form distance.

$$\sqrt{(\vec{p}-\vec{q})^{\mathrm{T}} \cdot \boldsymbol{V}_{\vec{p}} \cdot \boldsymbol{E'}_{\vec{p}} \cdot \boldsymbol{V}_{\vec{p}}^{\mathrm{T}} \cdot (\vec{p}-\vec{q})} \tag{24}$$

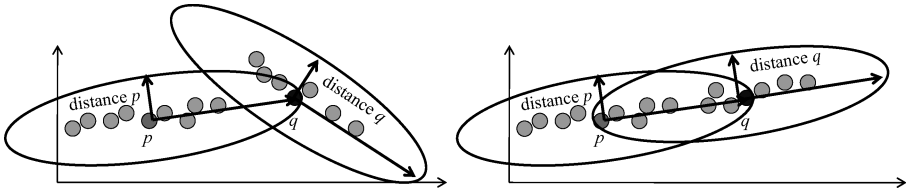


Fig. 19. 4C: distance between two points.

Applying this measure symmetrically to $\vec{q}$ and choosing the maximum of both distances helps to decide whether both points are connected by a similar correlation of attributes and, thus, are similar and belong to each other's correlation neighborhood. Figure 19 illustrates this idea. The ellipsoids represent the correlation neighborhoods of some sample objects. In the left example of Figure 19, p and q are not connected because q does not find p in its correlation neighborhood. On the righthand side, the points p and q are connected because they find one another in their correlation neighborhood.

As a hierarchical approach, HiCO [Achtert et al. 2006a] defines the distance between points according to their local correlation dimensionality and subspace orientation and uses hierarchical density-based clustering [Ankerst et al. 1999] to derive a hierarchy of correlation clusters.

COPAC [Achtert et al. 2007b] is based on similar ideas as 4C but disposes of some problems like meaningless similarity matrices due to sparse ε-neighborhoods, instead taking a fixed number k of neighbors, which raises the question how to choose a good value for k, but at least choosing $k > \lambda$ ensures a meaningful definition of a λ-dimensional hyperplane. The main point in COPAC, however, is a considerable speedup by partitioning the dataset based on the observation that a correlation cluster should consist of points exhibiting the same local correlation dimensionality (i.e., the same number of strong eigenvectors in the covariance matrix of the k-nearest neighbors). Thus, the search for clusters involves only the points with equal *local correlation dimensionality*. By creating one partition for each occurring correlation dimensionality, the time complexity rapidly decreases on average by getting rid of a squared factor d^2 in a d-dimensional dataset.

Another related algorithm is ERiC [Achtert et al. 2007a], also deriving a local eigensystem for a point based on the k-nearest neighbors in Euclidean space. Here, the neighborhood criterion for two points in a DBSCAN-like procedure is an approximate linear dependency and the affine distance of the correlation hyperplanes as defined by the strong eigenvectors of each point. Like in COPAC, the property of clusters to consist of points exhibiting an equal local correlation dimensionality is exploited for the sake of efficiency. Furthermore, the resulting set of clusters is also ordered hierarchically to provide the user with a hierarchy of subspace clusters. In finding and correctly assigning complex patterns of intersecting clusters, COPAC and ERiC improve considerably over ORCLUS and 4C.

Another approach based on PCA said to find even nonlinear correlation clusters, CURLER [Tung et al. 2005], seems not restricted to correlations of

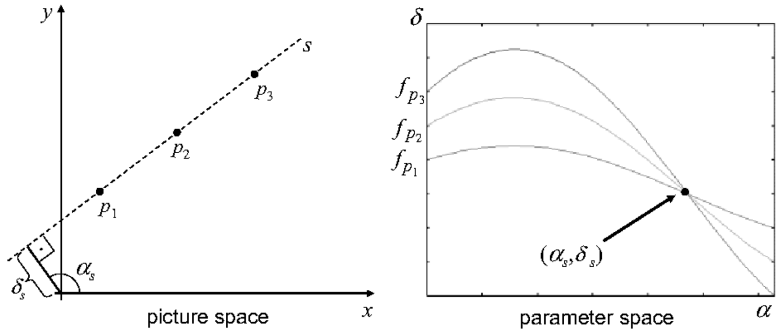


Fig. 20. Hough transform from picture space to parameter space using angle and radius parameters.

attributes but, according to its restrictions, finds any narrow trajectory and does not provide a model describing its findings.

PCA is a mature technique and allows the construction of a broad range of similarity measures grasping local correlation of attributes, and therefore finding arbitrarily oriented subspace clusters. A major intrinsic drawback common to all mentioned approaches is the notorious locality assumption. This assumption is widely accepted. But note that this innocent looking little (and often tacit) assumption boldly contradicts the basic problem statement: to find clusters in high-dimensional space that is doomed by the curse of dimensionality. To address problems occurring due to varying density in the local neighborhood, a framework for selecting a suitable neighborhood range and to stabilize the PCA by weighting the points has been proposed by Kriegel et al. [2008]. This framework allows to integrate all existing PCA-based correlation clustering approaches and it shows considerable enhancements in effectiveness. However, this is not the ultimate solution for problems of high-dimensional data spaces. As we will further discuss in more detail in Section 5, the curse of dimensionality condemns all distances to look alike, thus rendering nearest neighbor queries rather meaningless in high-dimensional data. Thus, to successfully employ PCA in correlation clustering in really high-dimensional data spaces may require even more effort henceforth.

4.2.2 *An Approach Based on the Hough Transform.* A completely different approach is pursued by the algorithm CASH [Achtert et al. 2008], based on the concepts of the Hough transform [Hough 1962; Duda and Hart 1972]. The Hough transform was originally designed to map the points from a two-dimensional data space (also called picture space) of Euclidean coordinates (e.g., pixels of an image) into a parameter space (see Figure 20). The parameter space represents all possible one-dimensional lines in the original two-dimensional, data space. In principle, each point of the data space is mapped into an infinite number of points in the parameter space, which is not materialized as an infinite set but instead as a trigonometric function in the parameter space. Each function in the parameter space represents all lines in the picture space crossing the corresponding point in data space. The intersection of two curves

in the parameter space indicates a line through both corresponding points in the picture space.

The objective of a clustering algorithm is to find intersections of many curves in the parameter space representing lines through many database objects. The key feature of the Hough transform is that the distance of the points in the original data space is not considered any more. Objects can be identified as associated to a common line, even if they are far apart in the original feature space. As a consequence, the Hough transform is a promising candidate for developing a principle for subspace analysis that does not require the locality assumption, thus enabling a global subspace clustering approach. CASH follows a grid-based approach to identify dense regions in the parameter space, successively attribute-wise dividing the space and counting the functions intersecting each of the resulting hyperboxes. In a depth-first search, most promising paths in the search tree are searched first. A hyperbox is divided along one axis if it contains enough functions to allow for dense child boxes in turn. If a dense subspace is found, the algorithm is applied on the dataset accounted for by the corresponding hyperbox projected on the corresponding subspace. This recursive descent allows for finding lower-dimensional subspace clusters and implicitly yields a hierarchy of arbitrarily oriented subspaces and their accommodated clusters. However, if there are no correlation clusters in the original data space and, hence, no dense regions in the parameter space (but still, the hyperboxes remain dense enough to qualify as promising candidates), the complete search space is enumerated resulting in a worst-case time complexity exponential in d. Probably, some more sophisticated heuristic may make this promising idea more practical for really high-dimensional data.

4.2.3 *Other Approaches.* Other basic principles that have been considered suitable for correlation analysis and for the search for arbitrarily oriented subspace clusters in the literature include the concepts of the *fractal dimension* and of *random sampling* methods.

—*Fractal Dimension.* There are some attempts to use the concept of self-similarity (fractal dimension) to cluster datasets [Barbara and Chen 2000; Parros Machado de Sousa et al. 2002; Gionis et al. 2005]. This would provide quite a different basis to grasping correlations in addition to PCA, and therefore constitutes a rather promising approach; however, it does assume the locality of patterns even by definition. Furthermore, the fractal dimension as a single property of a data (sub-)set does not yield information regarding the primary directions within a data distribution and, hence, seems less helpful for correlation clustering than PCA-based techniques. Among approaches based on these principles, there is seemingly no proposition mature enough to form the basis of a fully developed clustering procedure. So, it may require some effort to define really effective approaches to correlation clustering based on the fractal dimension. Nevertheless, we look forward to interesting new proposals in this field.

—*Random Sampling.* In the area of pattern recognition, a correlation clustering procedure based on random sampling has been proposed by Haralick

and Harpaz [2005]: An l-dimensional subspace as a cluster hyperplane (technically speaking, a linear manifold) is constructed by sampling $l + 1$ points. The sampling is repeated n times, where n is a threshold to ensure that with a certain probability at least the points obtained by one sample indeed belong to a common cluster. Deriving this kind of threshold, however, is based on a couple of simplifying assumptions, such as a rough estimate of the number of clusters and the points of the dataset being equally distributed among all existing clusters. Assigning the points to the corresponding subspace cluster allows to assess the discriminability allowed for by the current clustering. Starting with one-dimensional subspaces, the algorithm proceeds in a bottom-up manner. These considerations have the nice property that, in a sense, a correlation cluster model is fitted during the process.

Harpaz and Haralick [2007b] proposed a similar idea, this time based on RANSAC [Fischler and Bolles 1981], an established algorithm to find one-dimensional lines in a dataset by random sampling. The derived one-dimensional clusters are then refined by adding or removing features.

Unfortunately, in both cases, the experimental evaluation does not allow to estimate the merits of this approach in terms of efficiency or effectiveness in comparison to other correlation clustering approaches. Only a comparison with ORCLUS is presented by Haralick and Harpaz [2005], where the results are rather inconclusive. However, the authors present convincing formalizations of the problem, probably allowing for further generalizations. In summary, also in this approach we eagerly anticipate further enhancements.

4.3 Summary

As for biclustering approaches, a thorough evaluation of the different approaches in this field is still owing, especially in comparison to biclustering algorithms. Most biclustering approaches do not rely on the locality assumption, which makes them quite successful in many applications albeit their models are rather restricted compared to the model of correlation clusters. The idea to mine for objects exhibiting common patterns of correlations is pursued by both pattern-based and correlation clustering. While pattern-based approaches mine for the sole patterns of similar behavior of data objects, irrespective of their distance in Euclidean space, correlation clustering approaches widen the field of possible patterns to more complex, positive and negative correlations of attributes. The drawback of most correlation clustering approaches is their assumption that points of a cluster are still densely arranged in Euclidean space. Recent advances in correlation clustering, however, are starting to overcome the drawbacks of density-based approaches in high-dimensional data, in principle.

5. DISCUSSION

5.1 A Heuristic-Based Systematic View

The general approach to clustering in high-dimensional data seeks to find clusters in arbitrarily oriented subspaces of the data space. So the general objective is: "*Find a partitioning of the data where each cluster may exist in its own*

subspace." The partitioning need not be unique, that is, clusters may overlap. The subspaces may be axis-parallel or arbitrarily oriented and may or may not overlap.

Since clustering in general is an NP-hard problem [Slagle et al. 1975], efficient solutions always use heuristics to yield approximate solutions in efficient time and space requirements. These heuristics are reflected usually in different cluster models and different algorithmic approaches like locally optimizing algorithms, greedy search procedures, etc. The problem of finding the relevant subspaces is in fact an even more complex task. A general, naïve solution would examine all possible subspaces to look for clusters. Clearly, this is impossible in the general case, since the search space of all possible arbitrarily oriented subspaces is infinite. Thus, assumptions and heuristics are required to make feasible solutions possible.

Such assumptions and heuristics are, among others, as follows:

—the restriction of the search space to certain subspaces (e.g., axis-parallel subspaces) or certain patterns in the data matrix;

—a clustering criterion implementing the downward closure property;

—selecting one axis of the data matrix as "dimensions", the other one as "data objects";

—the locality assumption;

—the assumption of simple additive models (patterns);

—randomized, greedy procedures; and

—specifying the number of clusters in advance.

Some of these assumptions or heuristics are used to make the clustering problem feasible, others for coping with the subspace search problem. Since both problems need to be considered in a rather tightly integrated way, some of these assumptions and heuristics are even used for both problems.

All of the proposed methods are based on at least one assumption. Some corresponding properties of the most prominent algorithms are indicated in Table II.

Subspace or projected clustering restricts the search space to axis-parallel subspaces and makes use of a clustering criterion implementing the downward closure property (usually based on a global density threshold), or makes use of the locality assumption to enable efficient search heuristics. For example, we find most bottom-up approaches here (CLIQUE, ENCLUS, MAFIA, SUBCLU, and P3C) free from the locality assumption. Instead, they pursue a complete enumeration approach facilitated by an APRIORI-like search. Thus, they remain in $O(2^d)$ in the worst case.

Biclustering and pattern-based clustering approaches restrict the search space to special forms or locations of subspaces or half-spaces. If they include correlations among attributes (like δ-bicluster, FLOC, p-Cluster, MaPle, and CoClus), although they are free of the locality assumption, they are restricted to very special cases of correlations and pursue either a complete enumeration or require specification of the number of clusters in advance and perform a greedy search.

Table II. Properties of Clustering Algorithms

Algorithm	complex correlations	simple positive correlation	simple negative correlation	axis parallel	not relying on locality assumption	adaptive density threshold	independent with respect to order of attributes	independent with respect to order of objects	deterministic	arbitrary number of clusters	overlapping clusters	overlapping subspaces	simultaneously overlapping clusters and subspaces	arbitrary subspace dimensionality	hierarchical structure	avoiding complete enumeration	noise robust
axis parallel clustering																	
CLIQUE [Agrawal et al. 1998]				✓	✓		✓	✓	✓	✓	✓	✓	✓	✓			✓
ENCLUS [Cheng et al. 1999]				✓	✓		✓	✓	✓	✓	✓	✓	✓	✓			✓
MAFIA [Nagesh et al. 2001]				✓	✓		✓	✓	✓	✓	✓	✓	✓	✓			✓
SUBCLU [Kailing et al. 2004]				✓	✓		✓	✓	✓	✓	✓	✓	✓	✓			✓
PROCLUS [Aggarwal et al. 1999]				✓		✓						✓				✓	
PreDeCon [Böhm et al. 2004]				✓			✓	✓	✓	✓		✓				✓	✓
P3C [Moise et al. 2006]				✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓			✓
COSA [Friedman and Meulman 2004]				✓			✓	✓	✓			✓		✓		✓	✓
DOC [Procopiuc et al. 2002]				✓	✓		✓	✓		✓	✓	✓	✓	✓			
DiSH [Achtert et al. 2007]				✓	✓	✓		✓	✓	✓		✓		✓	✓	✓	✓
FIRES [Kriegel et al. 2005]				✓	✓	✓		✓	✓	✓	✓	✓	✓	✓		✓	✓
pattern-based clustering																	
Block clustering [Hartigan 1972]					✓	*n a*	✓	✓	✓					✓	✓		✓
δ-bicluster [Cheng and Church 2000]		✓		✓	✓	*n a*	✓	✓	✓		✓	✓		✓		✓	✓
FLOC [Yang et al. 2002]		✓		✓	✓	*n a*					✓	✓	✓	✓		✓	✓
p-Cluster [Wang et al. 2002]		✓		✓	✓	*n a*	✓	✓	✓	✓	✓	✓	✓	✓			✓
MaPle [Pei et al. 2003]		✓		✓	✓	*n a*	✓	✓	✓	✓	✓	✓	✓	✓			✓
CoClus [Cho et al. 2004]		✓		✓	✓	*n a*								✓		✓	
OP-Cluster [Liu and Wang 2003]					✓	*n a*	✓	✓	✓	✓	✓	*n a*	*n a*	*n a*			✓
correlation clustering																	
ORCLUS [Aggarwal and Yu 2000]	✓	✓	✓	✓			✓					✓				✓	
4C [Böhm et al. 2004a]	✓	✓	✓	✓			✓	✓	✓	✓		✓				✓	✓
COPAC [Achtert et al. 2007b]	✓	✓	✓	✓			✓	✓	✓	✓		✓		✓		✓	✓
ERiC [Achtert et al. 2007a]	✓	✓	✓	✓			✓	✓	✓	✓		✓		✓	✓	✓	✓
CASH [Achtert et al. 2008]	✓	✓	✓	✓	✓	*n a*		✓	✓	✓		✓		✓	✓		✓

Heuristics restricting the general problem (see Section 5.1) are here formulated negatively, that is, the more marks an algorithm features, the less assumptions are restricting the general search space (and, roughly, the more general is the algorithm).
Approaches neither marked as finding positive or negative correlations, nor axis parallel clusters, are specialized to certain patterns that constitute very particular subspaces.
n a: not applicable

While biclustering approaches often treat rows and columns interchangeably, axis-parallel subspace and projected clustering as well as correlation clustering approaches break this symmetry in favor of more efficient search heuristics, or based on and motivated by specific spatial intuitions. This asymmetry between columns and rows may also be related to the point of view of a database researcher, where columns correspond to attributes of a database entry and rows correspond to single database objects. Thus, in the context of a database, both types of information are addressed in a different way and it is assumed that the number of database entries (rows) by far exceeds the number of attributes (columns).

Almost all correlation clustering approaches proposed so far make use of the locality assumption but avoid complete enumeration. Hence they are efficient, but their effectiveness may be questionable. An approach not relying on the locality assumption is CASH [Achtert et al. 2008]. In turn, CASH suffers from the complete enumeration problem and remains rather inefficient. Another general way to get rid of the locality assumption is to base the clustering on a random sampling process. For these approaches, however, the quality of a retrieved clustering is a matter of luck.

5.2 A Problem-Oriented Systematic View

In Section 5.1, we gave an overview on the different algorithms, considering their different assumptions and heuristics to restrict the most general search space. Another possible point of view is based on the specific problems associated with high-dimensional data, commonly addressed all at once as the "curse of dimensionality."

5.2.1 *The Curse of Dimensionality in the Clustering Problem.* The term "curse of dimensionality" refers to a bundle of problems related to high-dimensional data spaces. In the following, we list those problems that are most relevant for clustering high-dimensional data.

—*Problem* 1. Bellman is often cited for the term "curse of dimensionality," which he describes as "a malediction that has plagued the scientists from earliest days" [Bellman 1961, p. 94]. However, Bellman merely describes the fact that more dimensions result in more possibilities of values and finally disable a complete enumeration approach simply because tabularization and visualization of functions becomes increasingly difficult or even impossible with more variables. This problem is mainly known in pattern recognition and is more elaborated in recent textbooks like Bishop [2006].

Of course, this problem relates to the clustering problem in general: Seeking a clustering of a dataset supposes the data is being generated by several functions. Ideally, a clustering model would enable the user to identify the functional dependencies resulting in the dataset at hand and, thus, to eventually find new and interesting insights in the laws of nature, economy, society, or whatever domain the dataset describes. These functions become more complex as more attributes contribute to the actual relationships.

—*Problem* 2. Concepts like proximity, distance, or neighborhood become less meaningful with increasing dimensionality of a dataset [Beyer et al. 1999; Hinneburg et al. 2000; Aggarwal et al. 2001]. Roughly, the results in these papers state that the relative distance of the farthest point and the nearest point converges to 0 for increasing dimensionality d. We have

$$\lim_{d \to \infty} \frac{dist_{\max} - dist_{\min}}{dist_{\min}} \to 0,$$

that is, discrimination between the nearest and the farthest neighbor becomes rather poor in high-dimensional space. This is by far a more fundamental problem than the mere performance degradation of algorithms on high-dimensional data.

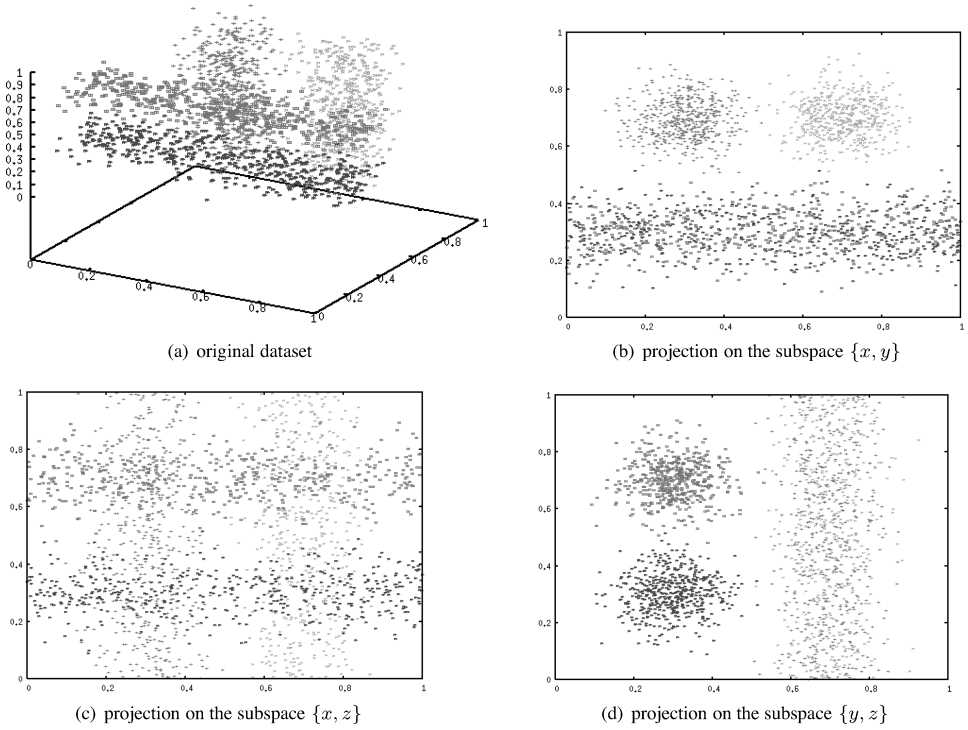
Clearly, the locality assumption is somewhat naïve in view of this problem. As a solution, a more deliberate choice of distance metrics (e.g., the use of Manhattan distance or even fractional distance metrics) has been proposed [Aggarwal et al. 2001]. However, this problem could be more fundamentally treated by dismissing any use of a local neighborhood in the clustering process.

It is important to note that these observations are valid for a broad range of data distributions and occur simply based on the mere number of dimensions. This problem is independent of the following problem, albeit the effect will be worsened considering Problem 3.

—*Problem* 3. In order to find dependencies and laws describing some occurring phenomena, a glut of data is collected and single entities are described with many possibly related attributes. Among these features, many irrelevant attributes can be expected. The relevance of certain attributes may differ for different groups of objects within the same dataset. Thus, since groups of data are defined by some of the available attributes only, many irrelevant attributes may interfere with the efforts to find these groups. Irrelevant attributes can also be related to as "noise." However, global feature reduction methods may be inadequate in face of the *local feature relevance* problem. If there is no (or not only) global noise but given sets of attributes are noisy only with respect to certain sets of objects, different clusters only exist in different subsets of attributes.

The challenge for clustering is therefore related to this problem: to find an appropriate subset of attributes to describe the similarity of objects belonging to the same group and possibly different subsets of attributes for different groups of objects. The cluster objects, then, reside in axis-parallel, affine subspaces of the complete data space.

Although we could expect, as a rule of thumb, the more irrelevant features in a dataset, the more dimensions there are at all, this problem can occur even in rather low-dimensional datasets, as can be seen in Figure 21: Four clusters in a three-dimensional dataset are characterized by two relevant attributes, each generated by a Gaussian distribution. The values in the remaining attribute are uniformly distributed in [0, 1] (see Figure 21(a)). For two clusters, however, the relevant attributes are x and y (Figure 21(b)) while for the remaining clusters, the relevant attributes are y and z (Figure 21(d)). Thus, these clusters can be discerned in the corresponding projections, and the remaining



(a) original dataset (b) projection on the subspace $\{x, y\}$

(c) projection on the subspace $\{x, z\}$ (d) projection on the subspace $\{y, z\}$

Fig. 21. A three-dimensional dataset illustrating problem 3.

clusters are intermixed. The third projection does also not allow a clear separation (Figure 21(c)). This problem has been elaborated in more detail by Parsons et al. [2004].

Many publications seem to obfuscate problems 2 and 3, but these are different effects in nature. However, irrespective of problem 2, distance measures may be seriously misguided by irrelevant attributes.

Let us note that related to problem 3 is the problem of subspace hierarchies, where lower-dimensional clusters are embedded in higher-dimensional ones; for example, in a three-dimensional space, a cluster with two relevant attributes (appearing as a line) may be embedded in a cluster with only one relevant attribute (appearing as a two-dimensional plane). While APRIORI-like bottom-up subspace search approaches usually implicitly compute these hierarchies (as long as the subspace clusters are axis-parallel), top-down approaches need explicitly be designed to generate information on the hierarchical relationships between the subspaces of clusters.

—*Problem* 4. Similarly as with problem 3, in a dataset containing many attributes, there may be some correlations among subsets of attributes. In sight of feature reduction methods, all but one of these attributes may be redundant. However, from the point of view of the domain scientist who collected these attributes in the first place, it may be an interesting new insight that there are so far unknown connections between features.

In view of spatial queries, the observation that the intrinsic dimensionality of a dataset is often lower than the embedding dimensionality (based on interdependencies among attributes) is often seen as a solution to overcome the curse of dimensionality [Faloutsos and Kamel 1994; Belussi and Faloutsos 1995; Pagel et al. 2000; Korn et al. 2001]. In view of the clustering problem, however, problems 1–3 remain unaffected by this phenomenon. In contrast, finding the correct subspace to define a suitable group of objects becomes an even harder problem, since cluster objects may reside in arbitrarily oriented, affine subspaces (due to problem 4); see Figure 15.

—*Other Problems.* There are quite a few other problems related to the curse of dimensionality. Most of these are, however, more relevant for indexing than for clustering data. Of course, clustering is affected by unbalanced range-queries largely searching in ranges beyond the boundaries of the dataset and similar problems, but these problems are elaborated extensively and in more detail in the literature concerned with index structures (e.g., see Lin et al. [1995], Berchtold et al. [1996, 2000, 1998a, 1998b]; Katayama and Satoh [1997]; Weber et al. [1998], Böhm and Kriegel [2000a, 2000b]). So we base our attempt of a systematic view on subspace clustering approaches on problems 1–4 as stated earlier.

5.2.2 *Approaches as Solutions to Specific Problems.* In view of the curse of dimensionality, the special cases described in Section 5.1 also reflect special problems.

While problem 1 plagues all approaches in general, the remaining problems are differently tackled individually or in combination by different approaches. A possible remedy for problem 2 is to avoid neighborhood queries. Thus, all approaches not relying on the locality assumption may be relatively unaffected by the problem of meaningless distance comparisons. Such approaches are especially bottom-up or hybrid approaches to axis-parallel subspace clustering and biclustering approaches (see Table II). Note, however, that Problem 2, as opposed to Problems 3 and 4, usually occurs in rather high-dimensional data but can also occur in moderate-dimensional data (i.e., $d = 10$–15). Problems 3 and 4, on the contrary, can occur in two-dimensional datasets already. It becomes just more likely that these problems occur by chance with increasing data dimensionality. It is therefore important not to mistake problem 2 for problem 3 or vice versa.

Problem 3 on its own leads to axis-parallel subspace clusters. So, all axis-parallel subspace and projected clustering approaches tackle especially this problem. Besides, this problem further worsens problem 2.

Biclustering approaches special forms of problem 4: simple positive correlations between all attributes in a subset of the attributes. Some take further assumptions into consideration. The strong point of these approaches is their being unconcerned about problem 2, since they generally do not take any distances between database objects into account.

Correlation clustering specifically tackles problem 4 in a general way. The occurrence of correlations among attributes alleviates problems 2 and 3 in a

way. However, with increasing dimensionality, we cannot help considering those problems, nevertheless.

The combination of problems 3 and 4 will result in clusters in very sparse, arbitrarily oriented subspaces. We expect the next generation of subspace clustering algorithms to address these problems in combination, simultaneously considering problem 2. A first attempt is presented by Achtert et al. [2008]; however, this attempt strikes problem 1, since this approach is trapped by the exponential time complexity of the complete enumeration in the worst case.

5.3 On the Difficulties in Solving Undefined Tasks

Among the multitude of algorithms, the task to solve remains often vague or undefined. It is often made clear which partial problems of the "curse of dimensionality" are tackled. But what remains unsaid is the exact meaning of the clusters retrieved by an algorithm. This vagueness is also an issue for comparatively evaluating different algorithms, as we will see later (Section 5.4). For now, we will concentrate on the obfuscation (in discussing axis-parallel subspace and projected clustering algorithms) resulting from confusing the two related, but obviously not identical, points of view for a classification of approaches: categorization according to the definition of the task versus categorization according to algorithmic aspects. As we will see, it is part of the problem that both points of view are indeed closely interrelated.

5.3.1 *Categorization With Respect to Task Definition.* The problem definition of projected clustering algorithms, namely to find a unique partitioning of points into clusters, is obviously heavily influenced by the traditional full-dimensional clustering problem. The only difference is that clusters may now exist in different subspaces of the original data, which makes the problem much harder to solve. However, in the context of many applications, it is sensible that points may be assigned to different clusters in different subspaces. Thus, not allowing any overlap of the clusters may be too strong a limitation in these applications.

On the other hand, finding *all* clusters in *all* subspaces is a rather arbitrary problem definition for two reasons. First, the number of clusters that are reported is usually very large. This overwhelmingly large set of clusters may quickly become too much for a user to analyze, interpret, and evaluate. Second, most of the clusters reported may be rather redundant because usually, any cluster in some k-dimensional subspace is also a cluster (or at least a subset of one) in all l-dimensional projections ($l < k$) of that subspace. In fact, the subspace clustering problem can be seen as a justification of a bottom-up subspace search strategy that delivers exactly the desired solution. However, the relevance of this problem statement is at least questionable and heavily depends on the clustering criterion. Most bottom-up algorithms rely on a density-based cluster model and have to apply a global density threshold for all subspaces in order to meet the downward closure property. As a consequence, it is often not clear how meaningful are the reported lower-dimensional subspace clusters. Recently, Moise and Sander [2008] discussed the problem of an overwhelmingly large number of redundant subspace clusters in a more formal way.

Hybrid approaches seem to offer a good deal between the limitations of the problem statements of projected clustering and subspace clustering. They usually allow overlapping clusters but do not overwhelm the user with the (partly redundant) glut of all clusters in all subspaces. However, it is often not clear what each single hybrid algorithm exactly searches for.

5.3.2 *Categorization With Respect to Algorithmic Aspects.* The problem of the top-down approaches is the circular dependency between cluster membership assignment and subspace learning from points of the cluster. In order to escape from this circular dependency, top-down approaches usually make the assumption that a subset of cluster members can be determined "somehow." In general, existing approaches implement two strategies for this "somehow": Many algorithms assume that the local neighborhood of cluster centers, or of other cluster members in the original full-dimensional feature space, contains a considerably large number of other cluster members (locality assumption). Other algorithms try to obtain cluster members from a random sample of points. The key point of these strategies is that the higher the number of outliers (non-cluster members) included in the selection, the less accurate the determination of the true subspace of the cluster. This has severe consequences because if the subspace of a cluster is not found correctly, again the assignment of points to this cluster may be less accurate. In sight of these considerations, both approaches (the locality assumption as well as random sampling) usually have rather strict limitations on the quality and applicability of the corresponding algorithms. A strong benefit of most top-down approaches is a good worst-case scalability. Usually, a complete enumeration of the exponential search space is avoided also in the worst case. Typically, algorithms implementing a top-down subspace search scale at most quadratically with respect to the dimensionality d of the data space.

In order to apply an efficient bottom-up subspace search approach similar to frequent itemset mining, the cluster criterion must implement the downward closure property. Existing bottom-up approaches usually rely on a density-based cluster criterion. A limitation of most of these approaches is that the cluster criterion must use a fixed density threshold for all subspaces in order to implement the downward closure property, although the use of a global density threshold is obviously contraindicated by the observations made before concerning the curse of dimensionality. As a consequence, the same globally defined density threshold applies for subspaces of considerably different dimensionality, although a significant cluster in a higher-dimensional subspace will most likely be less dense (in an absolute sense) than a significant cluster in a lower-dimensional subspace. In order to find higher-dimensional subspaces, the user has to define a less strict density threshold. This, however, would produce a lot of meaningless lower-dimensional clusters. On the other hand, choosing a more strict density threshold, the reported lower-dimensional clusters will probably be more meaningful but higher-dimensional subspace clusters will most likely be lost. The problem of applying a global density threshold obviously also affects the detection of meaningful subspace cluster hierarchies where lower-dimensional clusters are embedded in higher-dimensional ones. If the dimensionality of these

different subspace clusters significantly varies, a global density threshold will most likely miss important information on either the lower-dimensional clusters (in terms of false positives) or the higher-dimensional clusters (in terms of false negatives). Note that the term "meaningful" may be misleading: Of course, the clusters are meaningful in terms of the applied density threshold, but most likely the clusters do not make sense from an application point of view.

In addition, bottom-up subspace search is a complete enumeration approach, that is, the worst-case complexity is $O(2^d)$. On the average, usually the bottom-up search is also considerably less efficient than the top-down approach because in order to find a k-dimensional subspace, many one-dimensional, two-dimensional, ..., and $(k-1)$-dimensional subspaces need to be tested. As mentioned earlier, the fact that bottom-up approaches produce all lower-dimensional projections of subspaces accommodating clusters during the bottom-up traversal of the search space can be seen as an advantage. Thus, for addressing the subspace clustering problem, a bottom-up strategy would generally make sense in so far as the problem of meaningless lower-dimensional subspace clusters is concisely addressed.

5.3.3 *Summary.* In summary, a big problem in the field of finding clusters in axis-parallel subspaces is that existing papers usually lack a meaningful task definition. Very often, the problem statement is geared to the proposed algorithm, that is, the task that is to be solved is defined such that it matches the outcome of the proposed algorithm, as recently also pointed out by Moise and Sander [2008]. For example, the algorithm PreDeCon computes a DBSCAN-like partitioning of the data, but each cluster may exist in a different subspace. The problem statement of projected clustering perfectly matches this result: PreDeCon aims at finding a disjoint partitioning of points to clusters or noise and does not allow multiple cluster memberships. On the other hand, the algorithm CLIQUE computes all clusters in all subspaces using an APRIORI-style search. All subspace clusters generated during the APRIORI-like search are reported. Here, the problem statement of subspace clustering perfectly matches this result because all clusters in all subspaces are defined as relevant. As a consequence, the accuracy and applicability of methods for finding clusters in axis-parallel subspaces is hard to compare because each algorithm produces different clusters. This is not only because of different problem definitions, but also because of the application of different cluster models. When looking at the sample applications discussed before, these basic problem statements may be misleading for a user. For example, in gene expression analysis, the genes may explicitly be assigned to multiple clusters. On the other hand, any biologist will most likely be overwhelmed by inspecting the quantity and the redundancy of all clusters in all subspaces.

Obviously, the problem statement should dictate the algorithm's outcome and not vice versa. But as long as a meaningful general problem statement is missing, no algorithms can be designed to address it. By then, it is likely that further algorithms with slightly different problem statements will be proposed. However, an important contribution to the field would be to carefully analyze

relevant applications and to extract a meaningful problem definition that can be tackled.

5.4 Empirical Evaluation: A Desideratum

Newly proposed algorithms are often evaluated in a sloppy way, taking into account only one or two competitors (if at all), or even with a so-called "naïve" ad hoc solution for comparison of efficiency and effectiveness. Recently, an understanding for the need for consolidation of a maturing research area is rising in the research community, as illustrated by the discussions about the repeatability of results for SIGMOD 2008, the panel on performance evaluation at VLDB 2007, and the tentative special topic of "Experiments and Analyses Papers" at VLDB 2008. However, a fair and conclusive experimental evaluation of algorithms based on such a variety of assumptions and intuitions, pursuing such different search heuristics, and providing such a diversity of models and representations of results, seems rather difficult to achieve.

Aside from comparing apples and oranges in an evaluation of different algorithms, what would qualify as a good experimental evaluation? By all means, only stating that clusters can be found by a given approach is far too little because any partitioning algorithm like k-means always reports clusters. A solid evaluation needs to analyze the clusters in order to show that the grouping reflects some domain-specific knowledge. The usefulness of newly derived knowledge can usually only be interpreted by a domain expert who is most likely not at hand. Thus, showing that a clustering method can reproduce existing knowledge and does not produce implications that contradict existing knowledge will be a rather solid statement. In order to reproduce existing knowledge, it is possible to concentrate on labeled data. However, using labeled data in a clustering evaluation bears the risk of overfitting to the data and navigating into a supervised problem. Anyway, this is somehow frustrating because clustering as an unsupervised learning task aims at finding (beside already known information) previously unknown knowledge. On the other hand, if a (true) domain expert can be asked to interpret the results, this would be the *ne plus ultra* in reliability of the experiments. Unfortunately, the latter is most likely the rarest though most expedient and, thus, thrilling scenario. After all, this is why we should do data mining at all.

So far, there exists no complete competitive empirical evaluation with respect to efficiency and effectiveness of all or at least most of the prominent approaches. An attempt to provide a unified framework for all these algorithms is commenced by Achtert et al. [2008a].We sincerely hope that the systematic view and the theoretical comparison of different cluster models and objectives provided in this survey may be helpful for empirical studies in the future. However, a fair empirical evaluation of the different approaches is not a trivial task. The different heuristics and assumptions (refer to Section 5.1) and the different problems tackled (refer to Section 5.2) should always be kept in sight. In most cases, whether the trade-off between efficiency and effectiveness is tolerable will depend on the application.

6. CONCLUSIONS

Research in data mining and related disciplines such as statistics, pattern recognition, machine learning, and also applied sciences like, for example, bioinformatics, has led to a large variety of clustering techniques and addressed also the specialized problem of clustering high-dimensional data. New approaches to this problem are proposed in numerous conferences and journals every year. However, it is a well accepted opinion that there is no such thing as a general clustering technique suitable to all problems and universally applicable to arbitrary datasets. The aim of the concrete task of data analysis influences the choice of the clustering algorithm and obviously also the interpretation of the results of the clustering process.

The appropriate choice of a clustering approach adequate to the problem at hand should be based on knowledge of the basic principles the particular clustering approach is founded upon. Similarly, the interpretation of clustering results should be guided by the knowledge of the kinds of patterns a particular algorithm can or cannot find. In this survey, we aimed at supporting such decisions and interpretations by a systematic overview on the different kinds of algorithms specialized to different problems known to occur in high-dimensional data.

The family of axis-parallel subspace and projected clustering algorithms assumes that data objects belonging to the same cluster are near each other in Euclidean space, but allows to assess the corresponding distance of objects with respect to subsets of the attributes due to the problem of increasingly poor separation of near and far points in higher-dimensional data and the problem of irrelevant attributes. Pattern-based approaches often disregard the assumption that a cluster consists of objects that are near each other in the Euclidean space or some Euclidean subspace and, instead, aim at collecting objects following a similar behavioral pattern over a subset of attributes. These patterns relate to simple positive correlations among the considered attributes. Correlation clustering approaches generalize this approach to arbitrary complex positive or negative correlations but often assume, again, a certain density of the points in Euclidean space, as well. The general relationships between the different families of approaches are depicted in Figure 22. Let us note that the different notions of similarity employed by the different classes of algorithms surveyed in this study usually cannot be used interchangeably. Rather, the algorithms of each class are more or less tailored to the class-specific notions of similarity.

In identifying the problems addressed by the different families of algorithms and surveying the basic approaches pursued, we hope to stimulate further research. However, since different problems are addressed in different ways by the various approaches, questions arise whether more general solutions are possible addressing more problems in one approach. It has been suggested that the more general approach would be preferable and also gain the better results, since any structure imposed on the data would lead to biased results [Harpaz 2007; Harpaz and Haralick 2007a]. On the first glance, this seems to make sense. On the other hand, it is unclear whether this really is what potential users need. More than this: For supervised learning it is well known that a

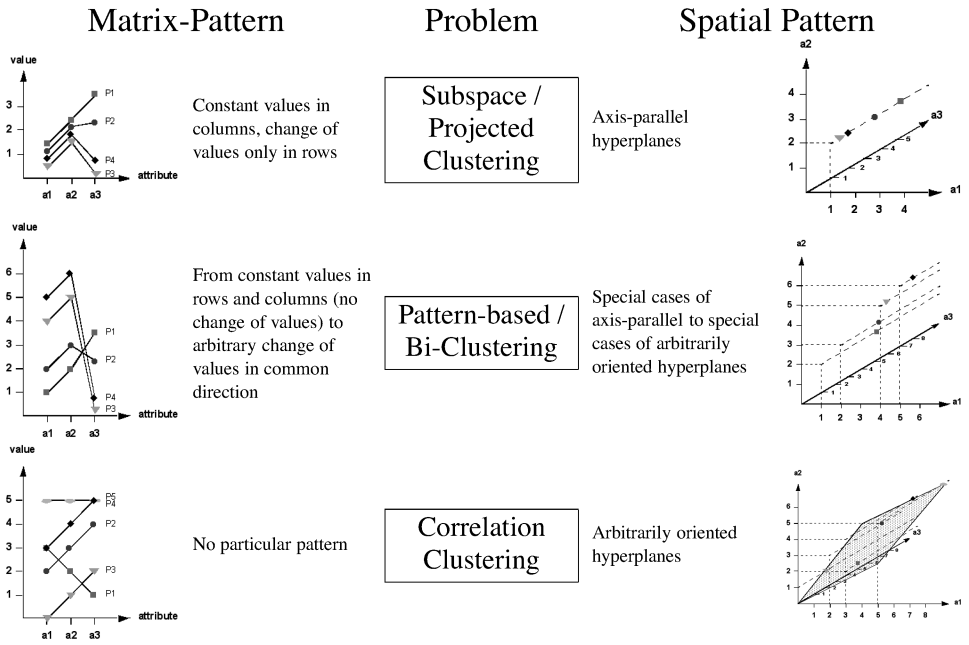


Fig. 22. The general relation between the major families of approaches.

bias-free learning is futile [Mitchell 1997]. In our opinion, in the context of unsupervised learning it remains to be seen whether the more general approach is *theoretically* the better one. There are even hints that the universal approach is impossible; see Faloutsos and Megalooikonomou [2007]. *Practically*, however, the domain expert should decide which bias (if any) is most meaningful in a given application. For example, it could be meaningful in a given application to specifically search for axis-parallel subspace clusters. This means, in turn, the use of algorithms such as P3C or of "soft" projected clustering algorithms would be discouraged in such a scenario, or otherwise the results are to be closely inspected, since some approaches can possibly result in arbitrarily oriented clusters although the algorithms are designed to search for axis-parallel ones (see Section 2.4).

In any case, a clustering approach seems to be more useful if it is able to provide a model describing the clusters found. This allows to understand the grouping, to identify underlying mechanisms quantitatively, and, eventually, to refine scientific theories or target marketing strategies. What is more, while defining a model is not possible without defining the task properly, a properly defined task will help to develop an optimization approach based on a suitable model.

Unfortunately, practitioners of cluster analysis usually are not able to use the approach that suits their purpose best, but only those approaches that are available in conveniently accessible statistical or data mining software systems. We totally agree with Kettenring [2008] that a more close cooperation of developers and practitioners of cluster analysis in order "to make sure that what is being offered represents best practices ... would be a much more valuable

contribution to cluster analysis than the next methodological advances." So, as a final remark, let us express the opinion that focusing on old or new problems in developing new clustering approaches should be done while even more closely analyzing the needs of potential applicants. The ultimate goal of clustering is to find new knowledge by analyzing datasets describing some aspects of the world. And eventually in these realms of applications, truth, as beauty, is in the eye of the beholder, or, as Hamlet said, "for there is nothing either good or bad, but thinking makes it so."

ACKNOWLEDGMENTS

We thank Gabriela Moise and Jörg Sander for valuable hints and beneficial discussions. We also wish to thank the attendees of the tutorial [Kriegel et al. 2007], this survey is based upon, where we presented the systematic of clustering approaches for the first time and received some helpful and encouraging remarks. We also wish to thank the audience of presentations of this tutorial at other international fora (PAKDD 2008, KDD 2008, and VLDB 2008), where questions and discussions again helped to further clarify the matter.

REFERENCES

ACHTERT, E., BÖHM, C., DAVID, J., KRÖGER, P., AND ZIMEK, A. 2008. Robust clustering in arbitrarily oriented subspaces. In *Proceedings of the 8th SIAM International Conference on Data Mining (SDM)*.

ACHTERT, E., BÖHM, C., KRIEGEL, H.-P., KRÖGER, P., MÜLLER-GORMAN, I., AND ZIMEK, A. 2007. Detection and visualization of subspace cluster hierarchies. In *Proceedings of the 12th International Conference on Database Systems for Advanced Applications (DASFAA)*.

ACHTERT, E., BÖHM, C., KRIEGEL, H.-P., KRÖGER, P., AND ZIMEK, A. 2006. Deriving quantitative models for correlation clusters. In *Proceedings of the 12th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD)*.

ACHTERT, E., BÖHM, C., KRIEGEL, H.-P., KRÖGER, P., AND ZIMEK, A. 2007a. On exploring complex relationships of correlation clusters. In *Proceedings of the 19th International Conference on Scientific and Statistical Database Management (SSDBM)*.

ACHTERT, E., BÖHM, C., KRIEGEL, H.-P., KRÖGER, P., AND ZIMEK, A. 2007b. Robust, complete, and efficient correlation clustering. In *Proceedings of the 7th SIAM International Conference on Data Mining (SDM)*.

ACHTERT, E., BÖHM, C., KRÖGER, P., AND ZIMEK, A. 2006a. Mining hierarchies of correlation clusters. In *Proceedings of the 18th International Conference on Scientific and Statistical Database Management (SSDBM)*.

ACHTERT, E., KRIEGEL, H.-P., AND ZIMEK, A. 2008a. ELKI: A software system for evaluation of subspace clustering algorithms. In *Proceedings of the 20th International Conference on Scientific and Statistical Database Management (SSDBM)*.

AGGARWAL, C. C., HINNEBURG, A., AND KEIM, D. 2001. On the surprising behavior of distance metrics in high dimensional space. In *Proceedings of the 8th International Conference on Database Theory (ICDT)*.

AGGARWAL, C. C., PROCOPIUC, C. M., WOLF, J. L., YU, P. S., AND PARK, J. S. 1999. Fast algorithms for projected clustering. In *Proceedings of the ACM International Conference on Management of Data (SIGMOD)*.

AGGARWAL, C. C. AND YU, P. S. 2000. Finding generalized projected clusters in high dimensional space. In *Proceedings of the ACM International Conference on Management of Data (SIGMOD)*.

AGRAWAL, R., GEHRKE, J., GUNOPULOS, D., AND RAGHAVAN, P. 1998. Automatic subspace clustering of high dimensional data for data mining applications. In *Proceedings of the ACM International Conference on Management of Data (SIGMOD)*.

AGRAWAL, R. AND SRIKANT, R. 1994. Fast algorithms for mining association rules. In *Proceedings of the ACM International Conference on Management of Data (SIGMOD)*.

ANKERST, M., BREUNIG, M. M., KRIEGEL, H.-P., AND SANDER, J. 1999. OPTICS: Ordering points to identify the clustering structure. In *Proceedings of the ACM International Conference on Management of Data (SIGMOD)*.

ASSENT, I., KRIEGER, R., MÜLLER, E., AND SEIDL, T. 2007a. DUSC: Dimensionality unbiased subspace clustering. In *Proceedings of the 7th International Conference on Data Mining (ICDM)*.

ASSENT, I., KRIEGER, R., MÜLLER, E., AND SEIDL, T. 2007b. VISA: Visual subspace clustering analysis. *ACM SIGKDD Explor. Newslett. 9*, 2, 5–12.

BANSAL, N., BLUM, A., AND CHAWLA, S. 2004. Correlation clustering. *Mach. Learn. 56*, 89–113.

BARBARA, D. AND CHEN, P. 2000. Using the fractal dimension to cluster datasets. In *Proceedings of the 6th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD)*.

BELLMAN, R. 1961. *Adaptive Control Processes. A Guided Tour*. Princeton University Press.

BELUSSI, A. AND FALOUTSOS, C. 1995. Estimating the selectivity of spatial queries using the 'correlation' fractal dimension. In *Proceedings of the 21st International Conference on Very Large Data Bases (VLDB)*.

BEN-DOR, A., CHOR, B., KARP, R., AND YAKHINI, Z. 2002. Discovering local structure in gene expression data: The order-preserving submatrix problem. In *Proceedings of the 6th Annual International Conference on Computational Molecular Biology (RECOMB)*.

BERCHTOLD, S., BÖHM, C., JAGADISH, H. V., KRIEGEL, H.-P., AND SANDER, J. 2000. Independent quantization: An index compression technique for high-dimensional data spaces. In *Proceedings of the 16th International Conference on Data Engineering (ICDE)*.

BERCHTOLD, S., BÖHM, C., AND KRIEGEL, H.-P. 1998. The pyramid technique: Towards breaking the curse of dimensionality. In *Proceedings of the ACM International Conference on Management of Data (SIGMOD)*.

BERCHTOLD, S., ERTL, B., KEIM, D. A., KRIEGEL, H.-P., AND SEIDL, T. 1998a. Fast nearest neighbor search in high-dimensional spaces. In *Proceedings of the 14th International Conference on Data Engineering (ICDE)*.

BERCHTOLD, S., KEIM, D. A., AND KRIEGEL, H.-P. 1996. The X-tree: An index structure for high-dimensional data. In *Proceedings of the 22nd International Conference on Very Large Data Bases (VLDB)*.

BEYER, K., GOLDSTEIN, J., RAMAKRISHNAN, R., AND SHAFT, U. 1999. When is "nearest neighbor" meaningful? In *Proceedings of the 7th International Conference on Database Theory (ICDT)*.

BISHOP, C. M. 2006. *Pattern Recognition and Machine Learning*. Springer.

BOUVEYRON, C., GIRARD, S., AND SCHMID, C. 2007. High-dimensional data clustering. *Comput. Statist. Data Anal. 52*, 502–519.

BÖHM, C., KAILING, K., KRIEGEL, H.-P., AND KRÖGER, P. 2004. Density connected clustering with local subspace preferences. In *Proceedings of the 4th International Conference on Data Mining (ICDM)*.

BÖHM, C., KAILING, K., KRÖGER, P., AND ZIMEK, A. 2004a. Computing clusters of correlation connected objects. In *Proceedings of the ACM International Conference on Management of Data (SIGMOD)*.

BÖHM, C. AND KRIEGEL, H.-P. 2000a. Dynamically optimizing high-dimensional index structures. In *Proceedings of the 7th International Conference on Extending Database Technology (EDBT)*.

BÖHM, C. AND KRIEGEL, H.-P. 2000b. Efficient construction of large high-dimensional indexes. In *Proceedings of the 16th International Conference on Data Engineering (ICDE)*.

CALIFANO, A., STOLOVITZKY, G., AND TU, Y. 2000. Analysis of gene expression microarrays for phenotype classification. In *Proceedings of the 8th International Conference on Intelligent Systems for Molecular Biology (ISMB)*.

CHAKRABARTI, K. AND MEHROTRA, S. 2000. Local dimensionality reduction: A new approach to indexing high dimensional spaces. In *Proceedings of the 26th International Conference on Very Large Data Bases (VLDB)*.

CHENG, C. H., FU, A. W.-C., AND ZHANG, Y. 1999. Entropy-Based subspace clustering for mining numerical data. In *Proceedings of the 5th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD)*. 84–93.

CHENG, H., HUA, K. A., AND VU, K. 2008. Constrained locally weighted clustering. In *Proceedings of the 34nd International Conference on Very Large Data Bases (VLDB)*.

CHENG, Y. AND CHURCH, G. M. 2000. Biclustering of expression data. In *Proceedings of the 8th International Conference on Intelligent Systems for Molecular Biology (ISMB)*.

CHO, H., DHILLON, I. S., GUAN, Y., AND SRA, S. 2004. Minimum sum-squared residue co-clustering of gene expression data. In *Proceedings of the 4th SIAM International Conference on Data Mining (SDM)*.

DHILLON, I. S. 2001. Co-clustering documents and words using bipartite spectral graph partitioning. In *Proceedings of the 7th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD)*.

DOMENICONI, C., PAPADOPOULOS, D., GUNOPULOS, D., AND MA, S. 2004. Subspace clustering of high dimensional data. In *Proceedings of the 4th SIAM International Conference on Data Mining (SDM)*.

DUDA, R. O. AND HART, P. E. 1972. Use of the Hough transformation to detect lines and curves in pictures. *Commun. ACM 15*, 1, 11–15.

DUDA, R. O., HART, P. E., AND STORK, D. G. 2001. *Pattern Classification*, 2nd ed. John Wiley&Sons.

ESTER, M., KRIEGEL, H.-P., SANDER, J., AND XU, X. 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the 2nd ACM International Conference on Knowledge Discovery and Data Mining (KDD)*.

FALOUTSOS, C. AND KAMEL, I. 1994. Beyond uniformity and independence: Analysis of R-trees using the concept of fractal dimension. In *Proceedings of the ACM International Conference on Management of Data (SIGMOD)*.

FALOUTSOS, C. AND MEGALOOIKONOMOU, V. 2007. On data mining, compression, and Kolmogorov complexity. *Data Mining Knowl. Discov. 15*, 1, 3–20.

FISCHLER, M. A. AND BOLLES, R. C. 1981. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM 24*, 6, 381–395.

FRIEDMAN, J. H. AND MEULMAN, J. J. 2004. Clustering objects on subsets of attributes. *J. Royal Statist. Soc. Series B (Statistical Methodology) 66*, 4, 825–849.

GANTER, B. AND WILLE, R. 1999. *Formal Concept Analysis. Mathematical Foundations*. Springer.

GAREY, M. R. AND JOHNSON, D. S. 1979. *Computers and Intractability. A Guide to the Theory of NP-Completeness*. W. H. Freeman.

GEORGII, E., RICHTER, L., RÜCKERT, U., AND KRAMER, S. 2005. Analyzing microarray data using quantitative association rules. *Bioinf. 21*, 2, ii1–ii8.

GETZ, G., LEVINE, E., AND DOMANY, E. 2000. Coupled two-way clustering analysis of gene microarray data. *Proc. National Academy Sci. United States Amer. 97*, 22, 12079–12084.

GIONIS, A., HINNEBURG, A., PAPADIMITRIOU, S., AND TSAPARAS, P. 2005. Dimension induced clustering. In *Proceedings of the 11th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD)*.

HAN, J. AND KAMBER, M. 2001. *Data Mining: Concepts and Techniques*. Academic Press.

HAN, J. AND KAMBER, M. 2006. *Data Mining: Concepts and Techniques*, 2nd ed. Academic Press.

HAND, D., MANNILA, H., AND SMYTH, P. 2001. *Principles of Data Mining*. The MIT Press.

HARALICK, R. AND HARPAZ, R. 2005. Linear manifold clustering. In *Proceedings of the 4th International Conference on Machine Learning and Data Mining in Pattern Recognition (MLDM)*.

HARPAZ, R. 2007. Model-Based linear manifold clustering. Ph.D. thesis, The City University of New York, Department of Computer Science.

HARPAZ, R. AND HARALICK, R. 2007a. Linear manifold correlation clustering. *Int. J. Inf. Technol. Intell. Comput. 2*, 2.

HARPAZ, R. AND HARALICK, R. 2007b. Mining subspace correlations. In *Proceedings of the IEEE Symposium on Computational Intelligence and Data Mining (CIDM)*.

HARTIGAN, J. A. 1972. Direct clustering of a data matrix. *J. Amer. Statist. Assoc. 67*, 337, 123–129.

HASTIE, T., TIBSHIRANI, R., AND FRIEDMAN, J. 2001. *The Elements of Statistical Learning. Data Mining, Inference, and Prediction*. Springer.

HINNEBURG, A., AGGARWAL, C. C., AND KEIM, D. A. 2000. What is the nearest neighbor in high dimensional spaces? In *Proceedings of the 26th International Conference on Very Large Data Bases (VLDB)*.

HOUGH, P. V. C. 1962. Methods and means for recognizing complex patterns. U.S. patent 3069654.
HUANG, J. Z., NG, M. K., RONG, H., AND LI, Z. 2005. Automated variable weighting in k-means type clustering. *IEEE Trans. Pattern Anal. Mach. Intell. 27*, 5, 657–668.
IHMELS, J., BERGMANN, S., AND BARKAI, N. 2004. Defining transcription modules using large-scale gene expression data. *Bioinf. 20*, 13, 1993–2003.
JAIN, A. K., MURTY, M. N., AND FLYNN, P. J. 1999. Data clustering: A review. *ACM Comput. Surv*.
JIANG, D., TANG, C., AND ZHANG, A. 2004. Cluster analysis for gene expression data: A survey. *IEEE Trans. Knowl. Data Eng. 16*, 11, 1370–1386.
JING, L., NG, M. K., AND HUANG, J. Z. 2007. An entropy weighting k-means algorithm for subspace clustering of high-dimensional sparse data. *IEEE Trans. Knowl. Data Eng. 19*, 8, 1026–1041.
JOLLIFFE, I. T. 2002. *Principal Component Analysis*, 2nd ed. Springer.
KAILING, K., KRIEGEL, H.-P., AND KRÖGER, P. 2004. Density-Connected subspace clustering for high-dimensional data. In *Proceedings of the 4th SIAM International Conference on Data Mining (SDM)*.
KATAYAMA, N. AND SATOH, S. 1997. The SR-tree: An index structure for high-dimensional nearest neighbor queries. In *Proceedings of the ACM International Conference on Management of Data (SIGMOD)*.
KETTENRING, J. R. 2008. A perspective on cluster analysis. Short communication. *Statist. Anal. Data Mining 1*, 1, 52–53.
KORN, F., PAGEL, B.-U., AND FALUTSOS, C. 2001. On the "dimensionality curse" and the "self-similarity blessing". *IEEE Trans. Knowl. Data Eng. 13*, 1, 96–111.
KRIEGEL, H.-P., KRÖGER, P., RENZ, M., AND WURST, S. 2005. A generic framework for efficient subspace clustering of high-dimensional data. In *Proceedings of the 5th International Conference on Data Mining (ICDM)*.
KRIEGEL, H.-P., KRÖGER, P., SCHUBERT, E., AND ZIMEK, A. 2008. A general framework for increasing the robustness of PCA-based correlation clustering algorithms. In *Proceedings of the 20th International Conference on Scientific and Statistical Database Management (SSDBM)*.
KRIEGEL, H.-P., KRÖGER, P., AND ZIMEK, A. 2007. Detecting clusters in moderate-to-high-dimensional data: Subspace clustering, pattern-based clustering, and correlation clustering. Tutorial at *the 7th International Conference on Data Mining (ICDM)*.
LI, J., HUANG, X., SELKE, C., AND YONG, J. 2007. A fast algorithm for finding correlation clusters in noise data. In *Proceedings of the 11th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD)*.
LIN, K., JAGADISH, H. V., AND FALOUTSOS, C. 1995. The TV-tree: An index structure for high-dimensional data. *VLDB J. 3*, 517–542.
LIU, B., XIA, Y., AND YU, P. S. 2000. Clustering through decision tree construction. In *Proceedings of the 9th International Conference on Information and Knowledge Management (CIKM)*.
LIU, G., LI, J., SIM, K., AND WONG, L. 2007. Distance based subspace clustering with flexible dimension partitioning. In *Proceedings of the 23th International Conference on Data Engineering (ICDE)*.
LIU, J. AND WANG, W. 2003. OP-cluster: Clustering by tendency in high dimensional spaces. In *Proceedings of the 3th International Conference on Data Mining (ICDM)*.
MADEIRA, S. C. AND OLIVEIRA, A. L. 2004. Biclustering algorithms for biological data analysis: A survey. *IEEE Trans. Comput. Biol. Bioinf. 1*, 1, 24–45.
MIRKIN, B. 1996. *Mathematical Classification and Clustering*. Kluwer.
MITCHELL, T. M. 1997. *Mach. Learn.*. McGraw-Hill.
MOISE, G. AND SANDER, J. 2008. Finding non-redundant, statistically significant regions in high dimensional data: A novel approach to projected and subspace clustering. In *Proceedings of the 14th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD)*.
MOISE, G., SANDER, J., AND ESTER, M. 2006. P3C: A robust projected clustering algorithm. In *Proceedings of the 6th International Conference on Data Mining (ICDM)*.
MOISE, G., SANDER, J., AND ESTER, M. 2008. Robust projected clustering. *Knowl. Inf. Syst. 14*, 3, 273–298.
MURALI, T. M. AND KASIF, S. 2003. Extracting conserved gene expression motifs from gene expression data. In *Proceedings of the 8th Pacific Symposium on Biocomputing (PSB)*.

NAGESH, H., GOIL, S., AND CHOUDHARY, A. 2001. Adaptive grids for clustering massive data sets. In *Proceedings of the 1st SIAM International Conference on Data Mining (SDM)*.

PAGEL, B.-U., KORN, F., AND FALOUTSOS, C. 2000. Deflating the dimensionality curse using multiple fractal dimensions. In *Proceedings of the 16th International Conference on Data Engineering (ICDE)*.

PARROS MACHADO DE SOUSA, E., TRAINA, C., TRAINA, A., AND FALOUTSOS, C. 2002. How to use fractal dimension to find correlations between attributes. In *Proceedings of the KDD-Workshop on Fractals and Self-Similarity in Data Mining: Issues and Approaches*.

PARSONS, L., HAQUE, E., AND LIU, H. 2004. Subspace clustering for high dimensional data: A review. *SIGKDD Explor. 6*, 1, 90–105.

PEI, J., ZHANG, X., CHO, M., WANG, H., AND YU, P. S. 2003. MaPle: A fast algorithm for maximal pattern-based clustering. In *Proceedings of the 3th International Conference on Data Mining (ICDM)*.

PFALTZ, J. 2007. What constitutes a scientific database? In *Proceedings of the 19th International Conference on Scientific and Statistical Database Management (SSDBM)*.

PRELIĆ, A., BLEULER, S., ZIMMERMANN, P., WILLE, A., BÜHLMANN, P., GUISSEM, W., HENNIG, L., THIELE, L., AND ZITZLER, E. 2006. A systematic comparison and evaluation of biclustering methods for gene expression data. *Bioinf. 22*, 9, 1122–1129.

PROCOPIUC, C. M., JONES, M., AGARWAL, P. K., AND MURALI, T. M. 2002. A Monte Carlo algorithm for fast projective clustering. In *Proceedings of the ACM International Conference on Management of Data (SIGMOD)*.

RÜCKERT, U., RICHTER, L., AND KRAMER, S. 2004. Quantitative association rules based on half-spaces: An optimization approach. In *Proceedings of the 4th International Conference on Data Mining (ICDM)* 507–510.

SEGAL, E., TASKAR, B., GASCH, A., FRIEDMAN, N., AND KOLLER, D. 2001. Rich probabilistic models for gene expression. *Bioinf. 17*, 1, S243–S252.

SEQUEIRA, K. AND ZAKI, M. J. 2005. SCHISM: A new approach to interesting subspace mining. *Int. J. Business Intell. Data Mining 1*, 2, 137–160.

SHENG, Q., MOREAU, Y., AND DE MOOR, B. 2003. Biclustering microarray data by Gibbs sampling. *Bioinf. 19*, 2, ii196–ii205.

SIM, K., LI, J., GOPALKRISHNAN, V., AND LIU, G. 2006. Mining maximal quasi-bicliques to cocluster stocks and financial ratios for value investment. In *Proceedings of the 6th International Conference on Data Mining (ICDM)*.

SLAGLE, J. L., CHANG, C. L., AND HELLER, S. L. 1975. A clustering and data-reorganization algorithm. *IEEE Trans. Syst. Man. Cybernetics* 5, 121–128.

TAN, P.-N., STEINBACH, M., AND KUMAR, V. 2006. *Introduction to Data Mining*. Addison Wesley.

TANAY, A., SHARAN, R., AND SHAMIR, R. 2002. Discovering statistically significant biclusters in gene expression data. *Bioinf. 18*, 1, S136–S144.

TANAY, A., SHARAN, R., AND SHAMIR, R. 2006. Biclustering algorithms: A survey. In *Handbook of Computational Molecular Biology*, S. Aluru, Ed. Chapman & Hall.

TUNG, A. K. H., XU, X., AND OOI, C. B. 2005. CURLER: Finding and visualizing nonlinear correlated clusters. In *Proceedings of the ACM International Conference on Management of Data (SIGMOD)*.

VAN MECHELEN, I., BOCK, H.-H., AND DE BOECK, P. 2004. Two-Mode clustering methods: A structured overview. *Statist. Methods Med. Res. 13*, 363–394.

WANG, H., WANG, W., YANG, J., AND YU, P. S. 2002. Clustering by pattern similarity in large data sets. In *Proceedings of the ACM International Conference on Management of Data (SIGMOD)*.

WEBB, G. I. 2001. Discovering associations with numeric variables. In *Proceedings of the 7th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD)*. 383–388.

WEBER, R., SCHEK, H.-J., AND BLOTT, S. 1998. A quantitative analysis and performance study for similarity-search methods in high-dimensional spaces. In *Proceedings of the 24th International Conference on Very Large Data Bases (VLDB)*.

WITTEN, I. H. AND FRANK, E. 2005. *Data Mining: Practical Machine Learning Tools and Techniques*, 2nd ed. Morgan Kaufmann.

WOO, K.-G., LEE, J.-H., KIM, M.-H., AND LEE, Y.-J. 2004. FINDIT: A fast and intelligent subspace clustering algorithm using dimension voting. *Inf. Softw. Technol. 46*, 4, 255–271.

YANG, J., WANG, W., WANG, H., AND YU, P. S. 2002. δ-clusters: Capturing subspace correlation in a large dataset. In *Proceedings of the 18th International Conference on Data Engineering (ICDE)*.

YIP, K. Y., CHEUNG, D. W., AND NG, M. K. 2004. HARP: A practical projected clustering algorithm. *IEEE Trans. Knowl. Data Eng. 16*, 11, 1387–1397.

YIP, K. Y., CHEUNG, D. W., AND NG, M. K. 2005. On discovery of extremely low-dimensional clusters using semi-supervised projected clustering. In *Proceedings of the 21st International Conference on Data Engineering (ICDE)*.

YIU, M. L. AND MAMOULIS, N. 2003. Frequent-Pattern based iterative projected clustering. In *Proceedings of the 3th International Conference on Data Mining (ICDM)*.

YIU, M. L. AND MAMOULIS, N. 2005. Iterative projected clustering by subspace mining. *IEEE Trans. Knowl. Data Eng. 17*, 2, 176–189.

Received May 2008; October 2008; accepted October 2008