

Ranking Interesting Subspaces for Clustering High Dimensional Data*

Karin Kailing, Hans-Peter Kriegel, Peer Kröger, and Stefanie Wanka

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

Abstract. Application domains such as life sciences, e.g. molecular biology produce a tremendous amount of data which can no longer be managed without the help of efficient and effective data mining methods. One of the primary data mining tasks is clustering. However, traditional clustering algorithms often fail to detect meaningful clusters because of the high dimensional, inherently sparse feature space of most real-world data sets. Nevertheless, the data sets often contain clusters hidden in various subspaces of the original feature space. We present a pre-processing step for traditional clustering algorithms, which detects all interesting subspaces of high-dimensional data containing clusters. For this purpose, we define a quality criterion for the interestingness of a subspace and propose an efficient algorithm called RIS (*Ranking Interesting Subspaces*) to examine all such subspaces. A broad evaluation based on synthetic and real-world data sets empirically shows that RIS is suitable to find all relevant subspaces in large, high dimensional, sparse data and to rank them accordingly.

1 Introduction

The tremendous amount of data produced nowadays in various application domains such as molecular biology can only be fully exploited by efficient and effective data mining tools. One of the primary data mining tasks is clustering which is the task of partitioning objects of a data set into distinct groups (clusters) such that two objects from one cluster are similar to each other, whereas two objects from distinct clusters are not.

Considerable work has been done in the area of clustering. Nevertheless, clustering real-world data sets often raises problems, since the data space is usually a high dimensional feature space. A prominent example is the application of cluster analysis to gene expression data. Depending on the goal of the application, the dimensionality of the feature space can be up to 10^2 when clustering

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

the genes and can be in the range of 10^3 to more than 10^4 when clustering the samples. In general, most of the common clustering algorithms fail to generate meaningful results because of the inherent sparsity of the data space. In such high dimensional feature spaces data does not cluster anymore. But usually, there are clusters in lower dimensional subspaces. In addition, objects can often be clustered differently in varying subspaces, i.e. objects may be grouped with different objects when subspaces vary. Again, gene expression data is a prominent example. When clustering the genes to detect co-regulated genes, one has to cope with the problem, that usually the co-regulation of the genes can only be detected in subsets of the samples (attributes). In other words, different subsets of the samples are responsible for different co-regulations of the genes. When clustering the samples this situation is even worse. As different phenotypes are hidden in varying subsets of the genes, the samples could usually be clustered differently according to various phenotypes, i.e. in varying subspaces.

1.1 Related Work

A common approach to cope with the curse of dimensionality for data mining tasks are dimensionality reduction or methods. In general, these methods map the whole feature space onto a lower-dimensional subspace of relevant attributes, using e.g. principal component analysis (PCA) and singular value decomposition (SVD). However, the transformed attributes often have no intuitive meaning any more and thus the resulting clusters are hard to interpret. In some cases, dimensionality reduction even does not yield the desired results (e.g. [1] presents an example where PCA does not reduce the dimensionality). In addition, using dimensionality reduction techniques, the data is clustered only in a particular subspace. The information of objects clustered differently in varying subspaces is lost. This is also the case for most common feature selection methods.

A second approach for coping with clustering high-dimensional data is projected clustering, which aims at computing k pairs $(C_i, S_i)_{(0 \leq i \leq k)}$ where C_i is a set of objects representing the i -th cluster, S_i is a set of attributes spanning the subspace in which C_i exists (i.e. optimizes a given clustering criterion), and k is a user defined integer. Representative algorithms include the k -means related PROCLUS [2], ORCLUS [3] and the density-based approach OPTIGRID [4]. While the projected clustering approach is more flexible than dimensionality reduction, it also suffers from the fact that the information of objects which are clustered differently in varying subspaces is lost. Figure 1(a) illustrates this problem using a feature space of four attributes A,B,C, and D. In the subspace AB the objects 1 and 2 cluster together with objects 3 and 4, whereas in the subspace CD they cluster with objects 5 and 6. Either the information of the cluster in subspace AB or in subspace CD will be lost.

The most informative approach for clustering high-dimensional data is subspace clustering which is the task of automatically identifying (in general several) subspaces of a high dimensional data space that allow better clustering of the data objects than the original space [1]. One of the first approaches to subspace clustering is CLIQUE [1], a grid-based algorithm using an *Apriori*-like method

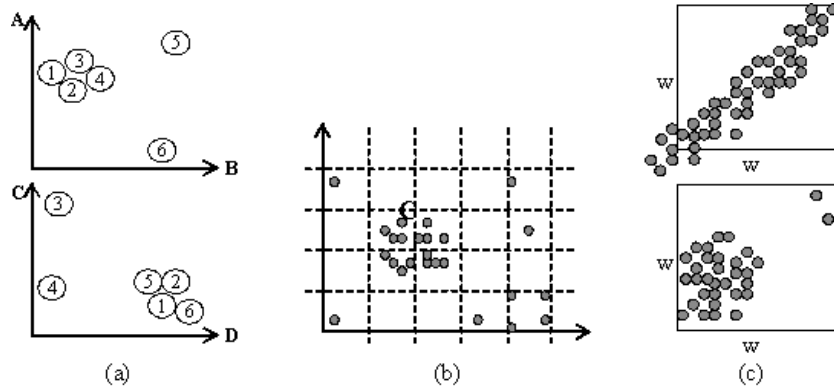


Fig. 1. Drawbacks of existing approaches (see text for explanation).

to recursively navigate through the set of possible subspaces in a bottom-up way. The dataspace is first partitioned by an axis-parallel grid into equi-sized blocks of width ξ called *units*. Only units whose densities exceed a threshold τ are retained. Both ξ and τ are the input parameters of CLIQUE. A cluster is defined as a maximal set of connected dense units. Successive modifications of CLIQUE include ENCLUS [5] and MAFIA [6]. But the information gain of these approaches is also sub-optimal. As they only provide clusters and not complete partitionings of some subspaces, we do not get the information in which subspaces the whole dataset clusters best. Another drawback of these methods is caused by the use of grids. In general, grid-based approaches heavily depend on the positioning of the grids. Clusters may be missed if they are inadequately oriented or shaped. Figure 1(b) illustrates this problem for CLIQUE: Each grid by itself is not dense, if $\tau > 4$, and thus, the cluster C is not found. On the other hand if $\tau = 4$, the cell with four objects in the lower right corner just above the x-axis is reported as a cluster.

Another recent approach called DOC [7] proposes a mathematical formulation for the notion of an optimal projected cluster, regarding the density of points in subspaces. DOC is not grid-based but as the density of subspaces is measured using hypercubes of fixed width w , it has similar problems drafted in Figure 1(c). If a cluster is bigger than the hypercube, some objects may be missed. Furthermore, the distribution inside the hypercube is not considered, and thus it need not necessarily contain only objects of one cluster.

1.2 Contributions

In this paper, we propose a new approach which eliminates the problems mentioned above and enables the user to gain all the clustering information contained in high-dimensional data. We present a preprocessing step, which selects all interesting subspaces using a density-connected clustering notion. Thus we are able to detect all subspaces containing clusters of arbitrary size and shape. We

first define the “interestingness” of subspaces in Section 2 and provide a quality criterion to rank the subspaces according to their interestingness. Afterwards any traditional clustering algorithm (e.g. the one the user is accustomed to) can be applied to these subspaces. In Section 3, we present an efficient density-based algorithm called RIS (*Ranking Interesting Subspaces*) for computing all those subspaces. A broad experimental evaluation of RIS based on artificial as well as on gene expression data is presented in Section 4. Section 5 draws conclusions.

2 Ranking Interesting Subspaces

2.1 Preliminary Definitions

Let DB be a data set of n objects with dimensionality d . We assume, that DB is a database of feature vectors ($DB \subseteq \mathbb{R}^d$). All feature vectors have normalized values, i.e. all values fall into $[0, attrRange]$ for a fixed $attrRange \in \mathbb{R}^+$. Let $\mathcal{A} = \{a_1, \dots, a_d\}$ be the set of all attributes a_i of DB . Any subset $S \subseteq \mathcal{A}$, is called a subspace. The projection of an object o into a subspace $S \subseteq \mathcal{A}$ is denoted by $\pi_S(o)$. The distance function is denoted by $dist$. We assume that $dist$ is one of the L_p -norms. The ε -neighborhood of an object o is defined by $\mathcal{N}_\varepsilon(o) = \{x \in DB \mid dist(o, x) \leq \varepsilon\}$. The ε -neighborhood of an object in a subspace $S \subseteq \mathcal{A}$ is denoted by $\mathcal{N}_\varepsilon^S(o) := \{x \in DB \mid dist(\pi_S(o), \pi_S(x)) \leq \varepsilon\}$.

2.2 Interestingness of a Subspace

Our approach to rate the interestingness of subspaces is based on a density-based notion of clusters. This notion is a common approach for clustering used by various clustering algorithms such as DBSCAN [8], DENCLUE [9], and OPTICS [10]. All these methods search for regions of high density in a feature space that are separated by regions of lower density. We adopt the notion of [8] to define “dense regions” by means of core-objects:

Definition 1. (*Core-Object*)

Let $\varepsilon \in \mathbb{R}$ and $MinPts \in \mathbb{N}$. An object o is called core object if $|\mathcal{N}_\varepsilon(o)| \geq MinPts$.

The core-object property is the key concept of the formal density-connected clustering notion in [8]. This property can also be used for deciding about the interestingness of a subspace. Obviously, if a subspace contains no core-object, it contains no dense region (cluster) and therefore contains no relevant information.

Observation 1. *The number of core-objects of a dataset DB (wrt. ε and $MinPts$) is proportional to the number of different clusters in DB and/or the size of the clusters in DB and/or the density of clusters in DB .*

This observation can be used to rate the interestingness of subspaces. However, simply counting all the core objects for each subspace delivers not enough information. Even if two subspaces contain the same number of core-objects the

quality may differ a lot. Dense regions contain objects which are no core-objects but lie within the ε -neighborhood of a core-object and are thus a vital part of the dense region. Therefore, it is not only interesting how many core-objects a subspace contains but also how many objects lie within the ε -neighborhood of these core-objects. In the following the variable $count[S]$ denotes the sum of all points lying in the ε -neighborhood of all core-objects in the subspace S . The number of core-objects of S is denoted by $core[S]$. If we measure the interestingness of a subspace according to its $count[S]$ value and rank all subspaces according to this quality value, two problems are not addressed. Since naturally with each dimension the number of expected objects in the ε -neighborhood of an object decreases, this naive quality value favors lower dimensional subspaces over higher dimensional ones. To overcome this problem we introduce a scaling coefficient that takes the dimensionality of the subspace into account. We take the ratio between the $count[S]$ value and the $count[S]$ value we would get if all data objects were uniformly distributed in S . For that purpose, we compute the volume of a d -dimensional ε -neighborhood denoted by Vol_ε^d and the number of objects lying in Vol_ε^d assuming uniform distribution.

Definition 2. *The quality of a subspace S , measuring the interestingness of S is defined by:*

$$\text{QUALITY}(S) = \frac{count[S]}{n \cdot \frac{Vol_\varepsilon^{dim[S]}.n}{attrRange^{dim[S]}}}$$

If $dist$ is the L_∞ -norm, Vol_ε^d is a hypercube and can be computed by $Vol_\varepsilon^d = (2\varepsilon)^d$, or if $dist$ is the Euclidian distance (L_2 -norm) Vol_ε^d is a hypersphere and can be computed as given below:

$$Vol_\varepsilon^d = \frac{\sqrt{\pi}^d}{\Gamma(d/2 + 1)} \cdot \varepsilon^d$$

where $\Gamma(x + 1) = x \cdot \Gamma(x)$, $\Gamma(1) = 1$ and $\Gamma(\frac{1}{2}) = \sqrt{\pi}$.

The second problem is the phenomenon that in high-dimensional spaces more and more points are located on the boundary of the data space. The ε -neighborhoods of these objects are smaller because they exceed the borders of the data space. In [11] the authors show that the average volume of the intersection of the data space and a hypersphere with radius ε can be expressed as the integral of a piecewise defined function integrated over all possible positions of the ε -neighborhood, i.e the core-objects. For our implementation we choose a less complex heuristics to eliminate this effect based on periodical extensions of the data space (cf. Section 3.2 for details).

For two arbitrary subspaces $U, V \in \mathbb{R}^d$ this quality criterion has two complementary effects which are summarized in the following observation:

Observation 2. *Let $U \supset V$. Then the following inequalities hold:*

1. $core[U] \leq core[V]$ and $count[U] \leq count[V]$.
2. If $core[U] = core[V]$ and $count[U] = count[V]$ then $\text{QUALITY}(U) > \text{QUALITY}(V)$.

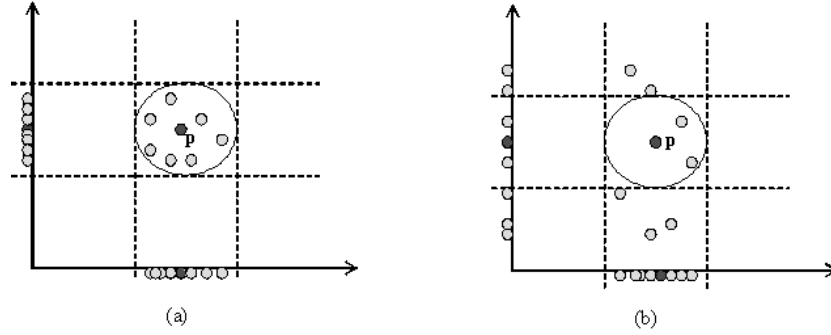


Fig. 2. Visualisation of Lemma 1 for $MinPts = 5$ (2D feature space).

The first observation states that, while navigating through the subspaces bottom-up, at a certain point the core-objects loose their core-object property due to the addition of irrelevant features and thus the quality decreases. On the other hand, as long as this is not the case, the features are relevant for the cluster and the quality increases.

2.3 General Idea of Finding Interesting Subspaces

A straightforward approach would be to examine all possible subspaces (e.g. bottom-up). The problem is, that the number of subspaces is 2^d . Basically all subspaces that do not contain any core-object can be dropped since they cannot contain any clusters. Furthermore, the core-object condition is decreasing strictly monotonic:

Lemma 1. (*Monotonicity of Core-Object Condition*)

Let $o \in DB$ and $S \subseteq \mathcal{A}$ be an attribute subset. If o is a core-object in S , then it is also a core-object in any subspace $T \subseteq S$ wrt. ε and $MinPts$, formally:

$$\forall T \subseteq S : |\mathcal{N}_\varepsilon^S(o)| \geq MinPts \Rightarrow |\mathcal{N}_\varepsilon^T(o)| \geq MinPts.$$

Proof. $\forall x \in \mathcal{N}_\varepsilon^S(o)$ the following holds:

$$\begin{aligned} dist(\pi_S(o), \pi_S(x)) \leq \varepsilon &\Rightarrow \sqrt[p]{\sum_{a_i \in S} (\pi_{a_i}(o) - \pi_{a_i}(x))^p} \leq \varepsilon \stackrel{T \subseteq S}{\Rightarrow} \\ \sqrt[p]{\sum_{a_i \in T} (\pi_{a_i}(o) - \pi_{a_i}(x))^p} \leq \varepsilon &\Rightarrow dist(\pi_T(o), \pi_T(x)) \leq \varepsilon \Rightarrow x \in \mathcal{N}_\varepsilon^T(o) \end{aligned}$$

It follows that $|\mathcal{N}_\varepsilon^T(o)| \geq |\mathcal{N}_\varepsilon^S(o)| \geq MinPts$ □

The Lemma is visualized in Figure 2(a). The reverse conclusion of Lemma 1 is illustrated in Figure 2(b) and states: If an object o is not a core-object in T , then o is also not a core-object in any super-space $S \supset T$.

The next sections will present in detail, how this property helps to eliminate a lot of subspaces in the process of generating all relevant subspaces in a bottom-up process.

```

RIS(SetOfObjects, Eps, MinPts)
  Subspaces := emptySet;
  FOR i FROM 1 TO SetOfObjects.size() DO
    Object := SampleObjects.get(i);
    RelevantSubspaces := GenerateSubspaces(Object, SetOfObjects);
    Subspaces.add(RelevantSubspaces);
  END FOR
  Subspaces.prune();
  Subspaces.sort();
END //RIS

```

Fig. 3. The RIS algorithm.

3 Implementation of RIS

3.1 Algorithm

Given a set of objects DB and density parameters ε and $MinPts$, RIS finds all interesting subspaces and presents them to the user ordered by relevance. For each object, RIS computes a set of relevant subspaces. All these sets are then merged. A pruning and sorting procedure is applied to the resulting set of subspaces. The pseudocode of the algorithm RIS is given in Figure 3. For each object $o \in DB$, all subspaces in which the core-object condition holds for o , are computed. This step will be described in detail in Section 3.2. Let us note that the algorithm can also be applied to a sample of DB , e.g. for performance reasons (cf. Section 4.3). For each detected subspace, statistical data is accumulated. The detected subspaces are pruned according to certain criteria. In Section 3.3, these criteria will be discussed. Finally, the subspaces are sorted for a more comprehensible user presentation. The clustering in these subspaces can then be done by any clustering algorithm.

3.2 Efficient Generation of Subspaces

For a given object $o \in DB$, the method `GenerateSubspaces` finds all subspaces S in which the core-object condition holds wrt. ε and $MinPts$. Formally, it computes the following set: $K_o := \{T \subseteq \mathcal{A} \mid |\mathcal{N}_\varepsilon^T(o)| \geq MinPts\}$.

The problem of finding the set K_o is equivalent to the problem of determining all frequent itemsets in the context of mining association rules [12] when using the L_∞ -norm as distance function and thus can be computed rather efficiently¹:

For each $x \in DB$ a transaction $T_x \subseteq \mathcal{A}$ is defined, such that,

$$a_i \in T_x \Leftrightarrow |\pi_{a_i}(x) - \pi_{a_i}(o)| \leq \varepsilon \quad \text{for all } i \in \{1, \dots, d\}.$$

¹ Let us note that the use of L_∞ -norm is no serious constraint. The only difference is that by using the L_∞ norm we may find additional core-objects and thus additional subspaces. However, these additional subspaces get low quality values anyway.

Lemma 2.

$$K_o = \{T \subseteq \mathcal{A} \mid \text{Supp}_{DB}(T) \geq \frac{\text{MinPts}}{|DB|}\} \text{ where } \text{Supp}_{DB}(T) = \frac{|\{x \in DB \mid T \subseteq T_x\}|}{|DB|}$$

Proof. $T \subseteq \mathcal{A} \wedge |N_\varepsilon^T(o)| \geq \text{MinPts}$

$$\Leftrightarrow T \subseteq \mathcal{A} \wedge |\{x \in DB \mid \text{dist}_{L_\infty}(\pi_T(o), \pi_T(x)) \leq \varepsilon\}| \geq \text{MinPts}$$

$$\Leftrightarrow T \subseteq \mathcal{A} \wedge$$

$$|\{x \in DB \mid \forall i \in \{1, \dots, d\} : a_i \in T \Rightarrow |\pi_{a_i}(o) - \pi_{a_i}(x)| \leq \varepsilon\}| \geq \text{MinPts}$$

$$\Leftrightarrow T \subseteq \mathcal{A} \wedge |\{x \in DB \mid T \subseteq T_x\}| \geq \text{MinPts} \Leftrightarrow T \subseteq \mathcal{A} \wedge \text{Supp}_{DB}(T) \geq \frac{\text{MinPts}}{|DB|} \quad \square$$

The method `GenerateSubspaces` extends the familiar *Apriori* [12] algorithm in accumulating the statistical information for measuring the subspace quality using the monotonicity of the core-object condition (cf. Lemma 1). As mentioned before, we are extending the data space periodically to ensure that all ε -neighborhoods have the same size. This can be done very easily by changing the way the transactions are defined. Instead of only checking if $|\pi_{a_i}(x) - \pi_{a_i}(o)| \leq \varepsilon$ we have to check if $|\pi_{a_i}(x) - \pi_{a_i}(o)| \leq \varepsilon$ or $|\pi_{a_i}(x) - \pi_{a_i}(o)| \geq \text{attrRange} - \varepsilon$.

3.3 Pruning of Subspaces

As we are only interested in the subspaces which provide the most information, we can perform the following downward pruning step to eliminate redundant subspaces: If there exists a $(k+1)$ -dimensional subspace S , with higher quality than the k -dimensional subspace T ($S \supset T$), we delete T .

For the second pruning, we assume, that for a given data set the k -dimensional subspace S reflects the clustering in that special data set in a best possible way. Thus, its quality value and the quality values of all its $(k-1)$ -dimensional subspaces T_1, \dots, T_m is high. On the other hand, if we combine one of these $(k-1)$ -dimensional subspaces T_1, \dots, T_m with another 1-dimensional subspace with lower quality, the quality of the resulting k -dimensional subspace can still be good. But as we know that it does not reflect the clustering in a best possible way, we are not interested in this k -dimensional subspace. The following heuristic upward pruning eliminates such subspaces. Let S be a k -dimensional attribute space and $S_{k-1} := \{T \mid T \subset S \wedge \dim[T] = k-1\}$ be the set of all $(k-1)$ -dimensional subspaces of S . Let $\overline{\text{count}}$ be the mean count value of all $T \in S_{k-1}$ and \bar{s} be the standard deviation. Let $\text{maxdiff} := \max_{T \in S_{k-1}} (|\text{count}[T] - \overline{\text{count}}|)$ be

the maximum deviation of the count-values of all $T \in S_{k-1}$ from the mean count-value. Then, the so-called *bias*-value can be computed as follows: $\text{bias} = \frac{\bar{s}}{\text{maxdiff}}$. If this bias-value falls below a certain threshold, we prune the k -dimensional subspace S . Experimental evaluations indicate that 0.56 is a good value for this bias-criterion.

3.4 Determination of Density Parameters

A heuristic method, which is experimentally shown to be sufficient, suggests $\text{MinPts} \approx \ln(n)$ where n is the size of the database. Then, ε must be picked

depending on the value of *MinPts*. In [8] a simple heuristic is presented to determine the ε of the "thinnest" cluster in the database (for a given *MinPts*). But as we do not know beforehand in which subspaces clusters will be found, we cannot determine ε to find a single subspace with one particular clustering. Quite the contrary, we want to choose the parameters such that RIS detects subspaces which might have clusters of different density and different dimensionality.

However, we can determine an upper bound for ε for a given value of *MinPts*. If we take uniform distribution as worst case, the ε -neighborhood of an object should not contain more than *MinPts* - 1 objects in the full-dimensional space. Otherwise all objects are core-objects. In case of the L_∞ -norm an upper bound for ε can be computed as follows:

$$n \cdot \frac{Vol_\varepsilon^d}{attrRange^{dim}} < MinPts \xrightarrow{L_\infty} \varepsilon < \frac{attrRange}{2} \cdot \sqrt[dim]{\frac{MinPts}{n}}$$

where $dim = d$. If we have any knowledge about the dimensionality of the subspaces we want to find, we can further decrease the upper bound by setting *dim* to the highest dimension of such a subspace.

This upper bound is very rough. Nevertheless, it provides a good indication for the choice of ε . Indeed, it empirically turned out, that *upperbound*/4 is a reasonable choice for ε . Experiments on synthetic data sets show, that our suggested criteria for the choice of the density parameters are sufficient to detect the relevant subspaces containing clusters.

4 Performance Evaluation

We tested RIS using several synthetic as well as a real-world data set. The experiments were run on a workstation with a 1.7 GHz CPU and 2 GB RAM.

The synthetic data sets were generated by a self-implemented data generator. It permits to control the size and structure of the generated data sets through parameters such as number and dimensionality of subspace clusters, dimensionality of the feature space and density parameters for the whole data set as well as for each cluster. In a subspace that contains a cluster the average density of data points in that cluster is much larger than the density of points not belonging to the cluster in this subspace. In addition, it is ensured, that none of the synthetically generated data sets can be clustered in full dimensional space.

The real world data set is the well-studied gene expression data set of Spellman et al. [13] analyzing the yeast mitotic cell cycle. We only chose the data of the *cdc15* mutant and eliminated all genes having missing attribute values. The resulting test data set consists of approximately 4400 genes expressed at 24 different time spots.

A subsequent clustering of the data sets in the detected subspaces was performed for each experiment using the above mentioned algorithm OPTICS to validate the interestingness of the subspaces computed by RIS.

4.1 Effectiveness Evaluation

Synthetic Data Sets. We evaluated the effectiveness of RIS using several synthetic data sets of varying dimensionality. The data sets contained between two and five overlapping clusters in varying subspaces. In all experiments, RIS detected the correct subspaces in which clusters exist and assigned the highest quality values to them. All higher dimensional subspaces which were generated, were removed by the upward pruning procedure.

Gene Expression Data. We also applied RIS to the above described gene expression data set. A clustering using OPTICS in the two top-ranked subspaces provided several clusters. The first subspace spanned by the time spots 90, 110, 130, and 190 contains three biologically relevant clusters with several genes playing a central role during mitosis². For example, cluster 1 consists of the genes CDC25 (starting point for mitosis), MYO3 and NUD1 (known for an active role during mitosis) and various other transcription factors (e.g. CHA4, ELP3) necessary during the cell cycle. Cluster 2 contains the gene STE12, identified by [13] as an important transcription factor for the regulation of the cell cycle. In addition, the genes CDC27 and EMP47 which have possible STE12-sites and are most likely co-regulated with STE12 are in that cluster. The cluster is completed by several transcription factors (e.g. XBP1, SSL1). Cluster 3 also consists of several genes which are known to play a role during the cell cycle such as DOM34, CKA1, CPA1, and MIP6. The second subspace is spanned by the time spots 190, 270 and 290 and consists of three clusters that have similar characteristics to those of the first subspace. In addition, a fourth cluster contains several mitochondrion related genes which have similar functions and are therefore most likely co-regulated, indeed. For example, the genes MRPL17, MRPL31, MRPL32, and MRPL33 are four mitochondrial large ribosomal subunits, the genes UBC1 and UBC4 are subunits of a certain protease, the genes SNF7 and VPS4 are direct interaction partners, and several other genes that code for mitochondrial proteins (e.g. MEF1, PHB1, CYC1, MGE1, ATP12). This indicates a higher mitochondrial activity at these time spots, which could be explained by a higher demand of biological energy during the cell cycle (the energy metabolism is located in mitochondrions). In summary, RIS detects two subspaces containing several biologically relevant co-regulations.

4.2 Efficiency Evaluation

The results of the efficiency evaluation are depicted in Figure 4. This evaluation is based on several synthetic data sets. The experiments were run with $MinPts = \ln(n)$ and ϵ chosen as suggested in Section 3.4. All run times are in seconds.

RIS scales well to the dimensionality of the relevant subspaces. With increasing dimensionality of the relevant subspaces, the runtime of RIS grows with a linear factor. On the other hand, the scalability of RIS to the size n and the

² The analysis of the clusters is partly based on the Saccharomyces Genome Database (SGD), available at: <http://genome-www.stanford.edu/Saccharomyces/>

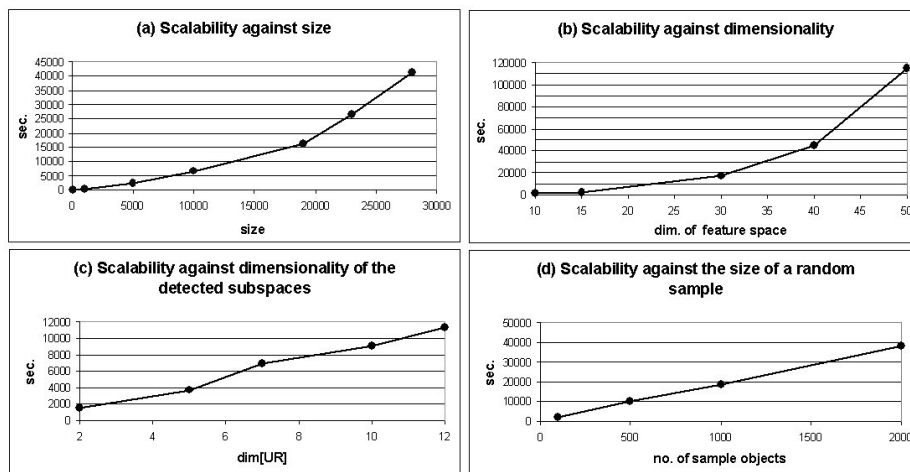


Fig. 4. Efficiency evaluation.

dimensionality d of the input data set is not linear. With increasing n and d , the runtime of RIS grows with an at least quadratic factor for rather large n and d , respectively. The reason for this scalability vs. the size n is that RIS performs multiple range-queries without any index support, due to the fact that the ε -neighborhoods of all points in arbitrary subspaces have to be computed. However, there is no index structure to efficiently support range queries in arbitrary subspaces. The observed scalability with respect to d can be explained by the *Apriori*-like navigation through the search space of all subspaces.

4.3 Speed-up for Large Data Sets

Since the runtime of RIS is rather high especially for large data sets, we applied random sampling to accelerate our algorithm. Figure 4 shows that for a large data set of $n = 750,000$ data objects, sampling yields a rather good speed-up. The data set contained two overlapping four-dimensional subspace clusters, containing approximately 400,000 and 350,000 points. Even using only 100 sample points, RIS had no problem to detect the subspaces of these two clusters. For all sample sizes, these subspaces had by far the highest quality values. Further experiments empirically show, that random sampling can be successfully applied to RIS in order to speed-up the runtime of this algorithm paying a minimum loss of quality.

5 Conclusions

In this paper, we introduced a preprocessing step for clustering high-dimensional data. Based on a quality criterion for the interestingness of a subspace, we presented an efficient algorithm called RIS to compute all interesting subspaces

containing dense regions of arbitrary shape and size. Furthermore, the well-established technique of random sampling can be applied to RIS in order to speed-up the runtime of the algorithm significantly with a minimum loss of quality. The effectiveness evaluation shows that RIS can be successfully applied to high-dimensional real-world data, e.g. on gene expression data in order to find co-regulated genes.

References

1. Agrawal, R., Gehrke, J., Gunopulos, D., Raghavan, P.: "Automatic Subspace Clustering of High Dimensional Data for Data Mining Applications". In: Proc. ACM SIGMOD Int. Conf. on Management of Data, Seattle, WA. (1998)
2. Aggarwal, C.C., Procopiuc, C.: "Fast Algorithms for Projected Clustering". In: Proc. ACM SIGMOD Int. Conf. on Management of Data, Philadelphia, PA. (1999)
3. Aggarwal, C., Yu, P.: "Finding Generalized Projected Clusters in High Dimensional Space". In: Proc. ACM SIGMOD Int. Conf. on Management of Data, Dallas, TX. (2000)
4. Hinneburg, A., Keim, D.: "Optimal Grid-Clustering: Towards Breaking the Curse of Dimensionality in High-Dimensional Clustering". In: Proc. 25th Int. Conf. on Very Large Databases, Edinburgh, Scotland. (1999)
5. Cheng, C.H., Fu, A.C., Zhang, Y.: "Entropy-Based Subspace Clustering for Mining Numerical Data". In: Proc. ACM SIGKDD Int. Conf. on Knowledge Discovery in Databases, San Diego, FL. (1999)
6. Goil, S., Nagesh, H., Choudhary, A.: "MAFIA: Efficient and Scalable Subspace Clustering for Very Large Data Sets". Tech. Report No. CPDC-TR-9906-010, Center for Parallel and Distributed Computing, Dept. of Electrical and Computer Engineering, Northwestern University (1999)
7. Procopiuc, C.M., Jones, M., Agarwal, P.K., Murali, T.M.: "A Monte Carlo Algorithm for Fast Projective Clustering". In: Proc. ACM SIGMOD Int. Conf. on Management of Data, Madison, WI. (2002) 418–427
8. Ester, M., Kriegel, H.P., Sander, J., Xu, X.: "A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise". In: Proc. 2nd Int. Conf. on Knowledge Discovery and Data Mining, Portland, OR. (1996) 291–316
9. Hinneburg, A., Keim, D.A.: "An Efficient Approach to Clustering in Large Multimedia Databases with Noise". In: Proc. 4th Int. Conf. on Knowledge Discovery and Data Mining, New York City, NY. (1998) 224–228
10. Ankerst, M., Breunig, M.M., Kriegel, H.P., Sander, J.: "OPTICS: Ordering Points to Identify the Clustering Structure". In: Proc. ACM SIGMOD Int. Conf. on Management of Data, Philadelphia, PA. (1999) 49–60
11. Berchtold, S., Böhm, C., Keim, D.A., Kriegel, H.P.: "A Cost Model For Nearest Neighbor Search in High-Dimensional Data Space". In: Proc. ACM PODS Symp. on Principles of Database Systems, Tucson, AZ. (1997) 78–86
12. Agrawal, R., Srikant, R.: "Fast Algorithms for Mining Association Rules". In: Proc. ACM SIGMOD Int. Conf. on Management of Data, Minneapolis, MN. (1994) 94–105
13. Spellman, P., Sherlock, G., Zhang, M., Iyer, V., Anders, K., Eisen, M., Brown, P., Botstein, D., Futcher, B.: "Comprehensive Identification of Cell Cycle-Regulated Genes of the Yeast *Saccharomyces Cerevisiae* by Microarray Hybridization.". *Molecular Biology of the Cell* **9** (1998) 3273–3297