Finding Hierarchies of Subspace Clusters

Elke Achtert, Christian Böhm, Hans-Peter Kriegel, Peer Kröger, Ina Müller-Gorman, and Arthur Zimek

Institute for Informatics, Ludwig-Maximilians-Universität München, Germany
{achtert, boehm, kriegel, kroegerp, muellerg, zimek}@dbs.ifi.lmu.de

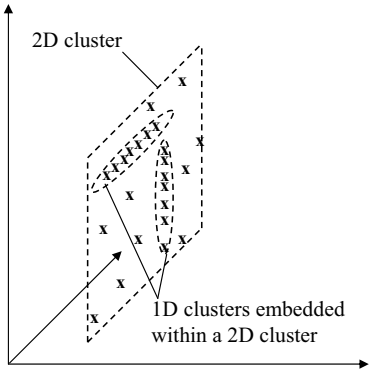
Abstract. Many clustering algorithms are not applicable to high-dimensional feature spaces, because the clusters often exist only in specific subspaces of the original feature space. Those clusters are also called subspace clusters. In this paper, we propose the algorithm HiSC (Hierarchical Subspace Clustering) that can detect hierarchies of nested subspace clusters, i.e. the relationships of lower-dimensional subspace clusters that are embedded within higher-dimensional subspace clusters. Several comparative experiments using synthetic and real data sets show the performance and the effectivity of HiSC.

1 Introduction

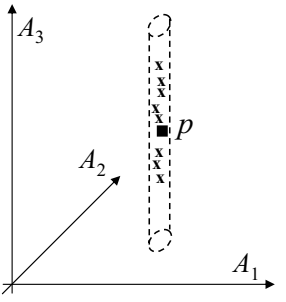
In high-dimensional feature spaces, many clustering algorithms are not applicable because the clusters often exist only in specific subspaces of the original feature space. This phenomenon is also often called *curse of dimensionality*. To detect such lower-dimensional subspace clusters, the task of subspace clustering (or projected clustering) has been defined recently. We will refer to a subspace cluster associated to a λ-dimensional projection/subspace (i.e. spanned by λ attributes) as a *λ-dimensional subspace cluster*. The dimensionality of a subspace associated to a subspace cluster is called *subspace dimensionality*.

In this paper, we focus on a second class of algorithms that assign each object to a unique cluster (or noise) rather than algorithms that allow overlapping subspace clusters. Existing algorithms for non-overlapping subspace clustering usually have one severe limitation in common. In case of hierarchically nested subspace clusters, i.e. several subspace clusters of low dimensionality may together form a larger subspace cluster of higher dimensionality, these algorithms will miss important information about the clustering structure. For example, consider two axis-parallel lines in a 3D space that are embedded into an axis-parallel 2D plane (cf. Figure 1(a)). Each of the two lines forms a 1-dimensional subspace cluster. On the other hand the plane is a 2-dimensional subspace cluster that includes the two 1-dimensional subspace clusters. In order to detect the lines, one has to search for 1-dimensional subspace clusters, whereas in order to detect the plane, one has to search for 2-dimensional subspace clusters. Moreover, searching subspace clusters of different dimensionality is essentially a hierarchical problem because the information that a point belongs e.g. to some k-dimensional subspace cluster that is itself embedded into an l-dimensional subspace cluster ($k < l$) can only be uncovered using a hierarchical approach.

J. Fürnkranz, T. Scheffer, and M. Spiliopoulou (Eds.): PKDD 2006, LNAI 4213, pp. 446–453, 2006.



(a) Hierarchically nested subspace clusters.



(b) Subspace dimensionality of a point (see text for details).

Several subspace clustering algorithms aim at finding all clusters in all subspaces of the feature space. Those algorithms produce overlapping clusters where one point may belong to different clusters in different subspaces. Well-known examples of such algorithms include e.g. CLIQUE [1], ENCLUS [2], SUBCLU [3], and FIRES [4].

Here, we focus on finding non-overlapping subspace clusters, i.e. assigning each point to a unique subspace cluster or noise. The probably most prominent examples for subspace clustering algorithms producing non-overlapping clusters are e.g. PROCLUS [5], DOC [6], and PreDeCon [7].

However, none of the proposed approaches to subspace clustering can detect nested hierarchies of subspace clusters. Thus, in this paper, we propose HiSC (Hierarchical Subspace Clustering), a new algorithm that applies a hierarchical approach to subspace clustering and, thus, detects hierarchies of subspace clusters.

2 Hierarchical Subspace Clustering

Let $\mathcal{D}$ be a data set of n feature vectors of dimensionality d ($\mathcal{D} \subseteq \mathbb{R}^d$). Let $\mathcal{A} = \{a_1, \ldots, a_d\}$ be the set of all attributes a_i of $\mathcal{D}$. Any subset $S \subseteq \mathcal{A}$, is called a *subspace*. The *projection* of an object $o \in \mathcal{D}$ 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 aim of HiSC is to detect clusters of lower dimensional subspaces contained in clusters of higher dimensional subspaces. Our general idea is to evaluate whether two points are contained in a common subspace cluster. For example, two points that are in a 1D subspace cluster may be contained in a 2D cluster that consists of the two 1D projections. We perfom this evaluation with a special distance measure called *subspace distance*. This distance results in a small value whenever two points are in a common low-dimensional subspace cluster, whereas the subspace distance is high if both points are in a common high-dimensional subspace cluster or are not in a subspace cluster at all. Therefore, our strategy is to merge those points into common clusters which have

small subspace distances. A hierarchy of subspace clusters means that clusters with small subspace distances are nested in clusters with higher subspace distances.

In order to define the already mentioned subspace distance, we assign a subspace dimensionality to each point of the database, representing the *subspace preference* of its local neighborhood. The subspace dimensionality of a point reflects those attributes having a small variance in the local neighborhood of the point. As local neighborhood of a point p we use the k-nearest neighbors of a point p, denoted by $\mathrm{NN}_k(p)$. The *variance* of the local neighborhood of a point $p \in \mathcal{D}$ from p along an attribute $A_i \in \mathcal{A}$, denoted by $\mathrm{VAR}_{A_i}(\mathrm{NN}_k(\mathrm{p}))$, is defined as follows:

$$\mathrm{VAR}_{A_i}(\mathrm{NN}_k(p)) = \frac{\sum_{q \in \mathrm{NN}_k(p)} (\pi_{A_i}(q) - \pi_{A_i}(p))^2}{|\mathrm{NN}_k(p)|}.$$

Intuitively, the subspace dimensionality is the number of attributes with high variance. Similar to [7], we assign a subspace preference vector to each point, indicating attributes with high and low variance.

Definition 1 (subspace preference vector of a point)
Let $\alpha \in \mathbb{R}$ be a threshold value. The subspace preference vector *of a point $p \in \mathcal{D}$, $w_p = (w_p^1, \ldots, w_p^d)^{\mathbf{T}}$, is defined as*

$$w_p^i = \begin{cases} 0 & \text{if} \quad \mathrm{VAR}_{A_i}(NN_k(p)) > \alpha \\ 1 & \text{if} \quad \mathrm{VAR}_{A_i}(NN_k(p)) \leq \alpha \end{cases}$$

The subspace dimensionality of a point can now be defined as follows.

Definition 2 (subspace dimensionality of a point)
The subspace dimensionality *λ_p of a point $p \in \mathcal{D}$ is the number of zero-values in the subspace preference vector of p, w_p, formally:*

$$\lambda_p = \sum_{i=1}^{d} \begin{cases} 1 & \text{if} \quad w_p^i = 0 \\ 0 & \text{if} \quad w_p^i = 1 \end{cases}$$

An example is visualized in Figure 1(b). The 9-nearest neighbors of the 3D point p exhibit a 1D subspace cluster spanned by the attribute A_3, i.e. the variance of the neighborhood of p is high along attribute A_3, whereas it is low along attributes A_1 and A_2. Consequently, $w_p = (1, 1, 0)^{\mathbf{T}}$ and $\lambda_p = 1$.

Once we have associated the points of our database to a (local) subspace dimensionality and to a subspace preference vector, we can now explain the main idea of our hierarchical clustering algorithm. Conventional hierarchical clustering algorithms like SLINK [8] or OPTICS [9] work as follows: They keep two separate sets of points, points which were already placed in the cluster structure and those which were not. In each step, one point of the latter set is selected and placed in the first set. The algorithm always selects that point which minimizes the distance to any of the points in the first set. By this selection strategy, the algorithm tries to extend the current cluster hierarchy as close to the bottom of the hierarchy as possible.

We will adapt this paradigm to the context of hierarchical subspace clustering where the hierarchy is a containment hierarchy of the subspaces. Two or more 1D subspace clusters may together form a 2D subspace cluster and so on. We simulate this behavior by defining a similarity measure between points which assigns a distance of 1, if these two points share a common 1D subspace cluster. If they share a common 2D subspace cluster, they have a distance of 2, etc. Sharing a common subspace cluster may mean different things: Both points may be associated to the same 2D subspace cluster, or both points may be associated to different 1D subspace clusters that intersect at some point or are parallel (but not skew).

If we assign a distance measure to a pair of points with the properties mentioned before, we can generally use the well-known hierarchical clustering algorithms. Intuitively, the distance measure between two points corresponds to the dimensionality of the data space which is spanned by the attributes of high variance of the neighborhoods of the two points. We first give a definition of the *subspace dimensionality of a pair of points* $\lambda(p, q)$ which follows the intuition of the spanned subspace. Then, we will define our subspace distance measure based on these concepts. In fact, the subspace dimensionality is the most important component of our distance measure.

Definition 3 (subspace preference vector/dimensionality of a pair of points)
The subspace preference vector $w(p, q)$ *of a pair of points* $p, q \in \mathcal{D}$ *representing the attributes with low and high variance of the combined subspace is defined by*

$$w(p, q) = w_p \wedge w_q \quad \textit{(attribute-wise logical AND-conjunction).}$$

The subspace dimensionality *between two points* $p, q \in \mathcal{D}$*, denoted by* $\lambda(p, q)$*, is the number of zero-values in* $w(p, q)$.

A first approach is defining the subspace distance between two points p and q as the subspace dimensionality $\lambda(p, q)$. We only need a slight extension for points that have the same subspace preference vector, but do not belong to the same subspace cluster. For these points, we have to check whether the preference vectors of two points are equal. If so, we have to determine the distance between the points along the attributes of low variance. If this distance, which can be evaluated by a simple weighted Euclidean distance using one of the preference vectors as weighting vector, exceeds α, the points (or corresponding clusters) do not belong to the same cluster but belong to different (parallel) clusters. The threshold α, playing already a key role in Definition 1, controls the degree of jitter of the subspace clusters.

As $\lambda(p, q) \in \mathbb{N}$, many distances between different point pairs are identical. Therefore, there are many tie situations during clustering. We resolve these tie situations by additionally considering the Euclidean distance within a subspace cluster as a secondary criterion. This means, inside a subspace cluster (if there are no nested lower-dimensional subspace clusters), the points are clustered in the same way as using a conventional hierarchical clustering method. The Euclidean distance between p and q hereby is weighted by the inverse of the combined preference vector $w(p, q)$ (as given in Definition 3). This inverse, $\bar{w}(p, q)$, weights the distance along attributes spanning the cluster with 1, the distance along any other attribute is weighted with 0. Formally we define:

Definition 4 (subspace distance)
Let $w = (w^1, \ldots, w^d)^{\mathbf{T}}$ *be a* d*-dimensional vector, and* $dist_w(p,q) = \sum w^i(\pi_{a_i}(p) - \pi_{a_i}(q))^2$ *be the weighted Euclidean distance w.r.t.* w *between two points* $p, q \in \mathcal{D}$*. The* subspace distance *between* p *and* q*, denoted by* $\text{SDIST}(p,q) = (d_1, d_2)$*, is a pair consisting of the following two values:*

$$d_1 = \lambda(p,q) + \begin{cases} 1 & \text{if } \max\{dist_{w_p}(p,q), dist_{w_q}(q,p)\} > \alpha \\ 0 & \text{else,} \end{cases}$$
$$d_2 = dist_{\bar{w}(p,q)}(p,q).$$

We say that $\text{SDIST}(p,q) \leq \text{SDIST}(r,s)$ *if* $\text{SDIST}(p,q).d_1 < \text{SDIST}(r,s).d_1$*, or* $\text{SDIST}(p,q).d_1 = \text{SDIST}(r,s).d_1$ *and* $\text{SDIST}(p,q).d_2 \leq \text{SDIST}(r,s).d_2$.

As discussed above, d_1 corresponds to the subspace dimensionality of p and q, taking special care in case of parallel clusters. The value d_2 corresponds to the weighted Euclidean distance between p and q, where we use the inverse of the combined preference vector, $\bar{w}(p,q)$, as weighting vector.

Using the subspace distance as defined in Definition 4 as a distance measure, we can basically run every hierarchical (or even non-hierarchical) clustering algorithm which is based on distance comparisons. Examples for such algorithms are Single-Link, Complete-Link, and the density-based method OPTICS [9]. HiSC computes a "walk" through the data set similar to OPTICS and assigns to each point o its smallest subspace distance with respect to a point visited before o in the walk. In each step of the algorithm, HiSC selects that point o having the minimum subspace distance to any already processed point. This process is managed by a seed list which stores all points that have been reached from already processed points sorted according to the minimum subspace distances. A special order of the database according to its subspace-based clustering structure is generated, the so-called *cluster order*, which can be displayed in a subspace distance diagram. Such a subspace distance diagram consists of the subspace distance values on the y-axis of all points, plotted in the order which HiSC produces on the x-axis. The result is a visualization of the clustering structure of the data set which is very easy to comprehend and interpret. The "valleys" in the plot represent the clusters, since points within a cluster typically have lower subspace distances than points outside of a cluster. The complete integration of our distance measure into the algorithm HiSC can be seen in Figure 1.

Input parameters. HiSC has two input parameters. First, the parameter k specifies the locality of the neighborhood from which the local subspace dimensionality of each point is determined. Obviously, this parameter is rather critical because if it is chosen too large, the local subspace preference may be blurred by noise points, whereas if it is chosen too small, there may not be a clear subspace preference observable, although existing. However, in our experiments, choosing k in the range between 10 and 20 turned out to produce very stable and robust results. Second, the parameter parameter α is important for specifying the attributes with low and high variance and, thus, α also affects the computation of the (local) subspace dimensionality of each point. In fact, attributes where the variance of the k-nearest neighbors of a point is below α are relevant for the subspace preference of the point. In our experiments, it turned out that

```
algorithm HiSC(𝒟, k, α)
    initialize empty priority queue pq   // ordered by SDIST
    for each p ∈ 𝒟 do
        compute w_p;
        p.SDIST = ∞;
        insert p into pq;
    while pq ≠ ∅ do
        o := pq.next();
        for each p ∈ pq do
            pq.update(p, SDIST(o, p));
        append o to the cluster order;
    return the cluster order;
```

Fig. 1. The HiSC algorithm

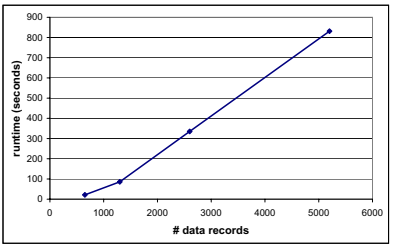
HiSC is quite robust against the choice of α, as long as α is chosen between 0.1% and 0.5% of the attribute range, i.e. the maximum attribute value. However, if one expects subspace clusters having a lot of jitter, α can be increased accordingly.

Runtime complexity. In the first loop the (local) subspace dimensionalities and preference vectors are precomputed which requires the determination of the k-nearest neighbors of each object. This step be done in $O(n \log n \cdot d)$ time assuming the use of a spatial index or in $O(n^2 \cdot d)$ withou index support. During the run of HiSC, we have to evaluate for each pair of points of the database its subspace dimensionality which is a simple logical AND-conjunction on the subspace dimensionality vectors and, thus, has a complexity of $O(d)$. Thus, the overall runtime complexity of HiSC is $O(n^2 \cdot d)$.

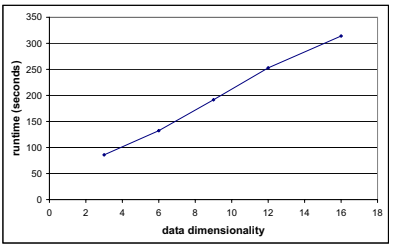
3 Experimental Evaluation

We evaluated the scalability of HiSC on a workstation featuring a 3.2 GHz CPU with 1GByte RAM. All parameters were choosen as suggested above. The scalability of HiSC w.r.t. the data set size is depicted in Figure 2(a). The experiment was run on a set of 3D synthetic data sets with varying number of records. Each data set contained two 1D clusters, two 2D clusters, and noise points. As it can be seen, HiSC scales nearly linearily w.r.t. the number of tuples in the dataset. A similar observation can be made when evaluating the scalability of HiSC w.r.t. the dimensionality of the data set (cf. Figure 2(b)). The experiments were obtained using data sets with 1,300 data points with varying dimensionality. Each data set contained two $(d-2)$-dimensional clusters, two $(d-1)$-dimensional clusters, and noise. Again, the result shows a linear increase of runtime when increasing the dimensionality of the data set.

We first evaluated HiSC on several synthetic data sets. Exemplary, we show the results on two data sets. Data set "DS1" (cf. Figure 3(a)) contains 3D points grouped in three hierarchical subspace clusters and noise. Two 1D clusters (cluster 1.1 and cluster 1.2) are both embedded within a 2D cluster (cluster 1). Data set "DS2" is a 20D data set containing three clusters of significantly different dimensionality and noise: cluster 1 is a 15D subspace cluster, cluster 2 is 10 dimensional, and cluster 3 is a 5D subspace cluster.

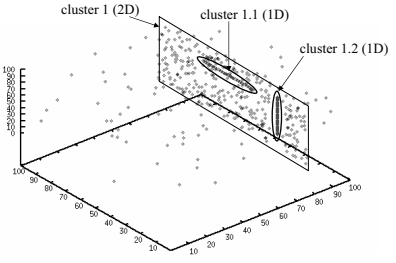


(a) Scalability w.r.t. size.

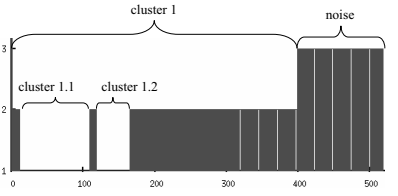


(b) Scalability w.r.t. dimensionality.

Fig. 2. Scalability of HiSC



(a) Data set.



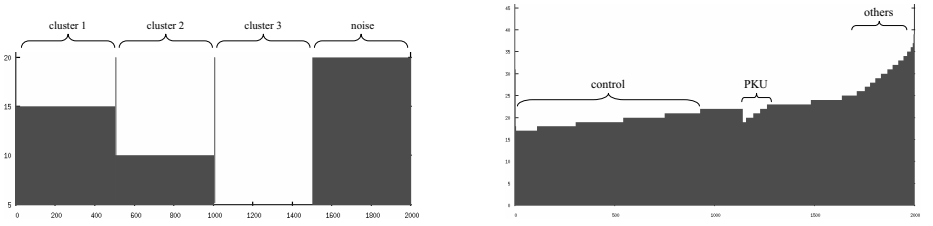
(b) Rechability diagram.

Fig. 3. Results on DS1 (3D)

The results of HiSC applied to DS1 are depicted in Figure 3(b). As it can be seen, the complete hierarchical clustering structure can be obtained from the resulting reachability plot. In particular, the nested clustering structure of the two 1D subspace clusters embedded within the 2D subspace cluster can be seen at first glance. Similar observations can be made when evaluating the reachability diagram obtained by HiSC on DS2 (cf. Figure 4(a)). HiSC has no problems with the three subspace clusters of considerably different dimensionality. The clusters can again be visually explored at first glance.

We also applied PreDeCon and PROCLUS on DS1 and DS2 for comparison. Neither PreDeCon nor PROCLUS are able to detect the hierarchies in DS1 and the subspace clusters of significantly different dimensionality.

We applied HiSC to a real-world data set containing metabolic screening data of 2,000 newborns. For each newborn, the blood-concentration of 43 different metabolites were measured. Thus, the data set is 43-dimensional containing 2,000 objects. The newborns are labeled by one of three categories. The healthy patients are marked by "control", newborns suffering phenylketonuria (a well-known metabolic disease) are labeled with "PKU", and newborns suffering any other metabolic disease are labeled with "others". The resulting reachability plot HiSC generates when applied to this data set is visualized in Figure 4(b). As it can be seen, HiSC produced a large hierarchy of 17D to 22D subspace clusters nested into each other. All these clusters contain approximately 98% newborns marked with "control". A second hierarchy of nested clusters contains only newborns marked with "PKU". The rest is a mix of all three categories.



(a) Results on DS2. (b) Results on metabolome data.

Fig. 4. Results on higher-dimensional data

4 Conclusions

In this paper, we presented HiSC, the first subspace clustering algorithm for detecting hierarchies of subspace clusters. HiSC scales linearly in the dimensionality of the data space and quadratically in the number of points. Several comparative experiments using synthetic and real data sets show that HiSC has a superior performance and effectivity compared to existing methods.

References

1. Agrawal, R., Gehrke, J., Gunopulos, D., Raghavan, P.: Automatic subspace clustering of high dimensional data for data mining applications. In: Proc. SIGMOD. (1998)
2. Cheng, C.H., Fu, A.W.C., Zhang, Y.: Entropy-based subspace clustering for mining numerical data. In: Proc. KDD. (1999)
3. Kailing, K., Kriegel, H.P., Kröger, P.: Density-connected subspace clustering for high-dimensional data. In: Proc. SDM. (2004)
4. Kriegel, H.P., Kröger, P., Renz, M., Wurst, S.: A generic framework for efficient subspace clustering of high-dimensional data. In: Proc. ICDM. (2005)
5. Aggarwal, C.C., Procopiuc, C.M., Wolf, J.L., Yu, P.S., Park, J.S.: Fast algorithms for projected clustering. In: Proc. SIGMOD. (1999)
6. Procopiuc, C.M., Jones, M., Agarwal, P.K., Murali, T.M.: A Monte Carlo algorithm for fast projective clustering. In: Proc. SIGMOD. (2002)
7. Böhm, C., Kailing, K., Kriegel, H.P., Kröger, P.: Density connected clustering with local subspace preferences. In: Proc. ICDM. (2004)
8. Sibson, R.: SLINK: An optimally efficient algorithm for the single-link cluster method. The Computer Journal **16** (1973) 30–34
9. Ankerst, M., Breunig, M.M., Kriegel, H.P., Sander, J.: OPTICS: Ordering points to identify the clustering structure. In: Proc. SIGMOD. (1999)