# What are clusters in high dimensions and are they difficult to find?

Frank Klawonn[1][2], Frank Höppner[1], and Balasubramaniam Jayaram[3]

[1] Department of Computer Science
Ostfalia University of Applied Sciences
Salzdahlumer Str. 46/48, D-38302 Wolfenbuettel, Germany
`{f.klawonn,f.hoeppner}@ostfalia.de`
[2] Bioinformatics & Statistics
Helmholtz Centre for Infection Research
Inhoffenstr. 7, D-38124 Braunschweig, Germany
`frank.klawonn@helmholtz-hzi.de`
[3] Department of Mathematics
Indian Institute of Technology Hyderabad
Yeddumailaram - 502 205, India
`jbala@iith.ac.in`

**Abstract.** The distribution of distances between points in a high-dimensional data set tends to look quite different from the distribution of the distances in a low-dimensional data set. Concentration of norm is one of the phenomena from which high-dimensional data sets can suffer. It means that in high dimensions – under certain general assumptions – the relative distances from any point to its closest and farthest neighbour tend to be almost identical. Since cluster analysis is usually based on distances, such effects must be taken into account and their influence on cluster analysis needs to be considered. This paper investigates consequences that the special properties of high-dimensional data have for cluster analysis. We discuss questions like when clustering in high dimensions is meaningful at all, can the clusters just be artifacts and what are the algorithmic problems for clustering methods in high dimensions.

## 1 Introduction

Clustering is an exploratory data analysis method applied to data in order to discover structures or groups – called clusters – in a data set. Data objects within the same cluster should be similar, data objects from different clusters dissimilar. This means that cluster analysis must be based on a similarity or nearness concept to measure how close or similar data objects are. Often a dual concept to similarity or nearness – a distance measure – is used. Especially, for data sets having exclusively real-valued attributes, i.e. data sets that are subsets of $\mathbb{R}^m$ where $m$ is the dimension of the data set, the Euclidean metric or – more generally – any metric derived from a norm on $\mathbb{R}^m$ can be seen as a candidate of a distance measure on which clustering can be based.

It should be noted that it is usually recommended to carry out some kind of normalisation on the single dimensions to avoid that the measurement units of the single dimensions influence the distances and therefore also the clustering results. If, for instance,

one of the attributes would be the height of persons, the data could be represented in various measurement units like centimetres, metres or also millimetres. The measurement of the height in metres would mean that the difference of two "data objects" (persons) in this attribute will normally be less than 1. If, however, the height is measured in millimetres, a difference of 100 between two persons would not be unusual. There are various strategies to normalise a real-valued attribute, for example $z$-score normalisation where the mean value is subtracted from each value and then a division by the sample standard deviation of the corresponding attribute is carried out. However, normalisation is not the topic of this paper and we refer to the overview on normalisation in [1].

In this paper, we focus on clustering high-dimensional data having only real-valued attributes.

The term *curse of dimensionality* was coined by Bellman [2], referring to the combinatorial explosion that is often implied by handling a large number of dimensions. In the context of cluster analysis, for many algorithms the number of dimensions does not cause serious computational problems. The number of data objects is usually more critical from the computational complexity point of view than the number of dimensions.

Today, the term curse of dimensionality is understood in more general terms, covering also other aspects of high-dimensional data than just problems of computational complexity. Distance measures like the Euclidean distance for high-dimensional data exhibit surprising properties that differ from what is usual for low-dimensional data. Two examples for such properties are the *concentration of norm phenomenon* – stating that under certain assumptions the relative distances from any point to its closest and farthest neighbour tend to be almost identical for high-dimensional data – and the *hubness phenomenon* where it seems that for high-dimensional data the distribution of the number of times a data point occurs among the $k$ nearest neighbours of other data points is extremely skewed to the right and a few data points – the hubs – are found very often among the $k$ nearest neighbours of other data points.

But what are the consequences of these effects for cluster analysis? In order to find answers to this question, we have to take a closer look at these phenomena in Section 2 and we also need a brief overview on clustering approaches in Section 3. In Section 4, we relate properties of high-dimensional data to clustering and discuss the crucial question what a cluster in high dimensions is. Based on these considerations, we consider the consequences for clustering high-dimensional data in terms of what is meaningful and in terms of algorithmic problems in Section 5. We summarise the results of the paper in Section 6.

## 2   Properties of high-dimensional data

The *concentration of norm phenomenon* (CoN) can formally be described in the following way [3]. Let $X_m$ be an $m$-dimensional random vector and let $d_m(x)$ denote the distance of $x \in \mathbb{R}^m$ to the origin of the coordinate system based on a suitable distance measure, for instance the Euclidean distance. Let $n \in \mathbb{N}$ be the size of the sample that is taken from the random vector $X_m$. Let $d_m^{(\text{max})}$ and $d_m^{(\text{min})}$ denote the largest and the

smallest distance of a point in the sample to the origin of the coordinate system. Then

$$\lim_{m \to \infty} \mathrm{Var}\left( \frac{d_m(X_m)}{\mathrm{E}(d_m(X_m))} \right) = 0 \;\Rightarrow\; \frac{d_m^{(\max)} - d_m^{(\min)}}{d_m^{(\min)}} \to_p 0 \qquad (1)$$

holds, where $\to_p$ denotes convergence in probability. In other words, when the relative variance – relative with respect to the mean distance – of the distances to the origin converges to zero for higher dimensions, then the relative difference of the closest and farthest point in the data set goes to zero with increasing dimensions. The requirement that the relative variance goes to zero is for instance satisfied when the random vector $X_m$ is a sample from $m$ independent and identically distributed random variables with finite expectation and variance and the Euclidean distance is used as the distance measure.

The converse theorem also holds [4].

It should be noted that the choice of the origin of the coordinate system as the query point to which the distances of the data points are computed is not of importance. Equation (1) is also valid for any other query or reference point. The same applies to the distance measure. A detailed investigation on fractional distances is provided in [5]. A discussion on various distance measures in connection with the CoN phenomenon is provided in [6–8].

The *hubness phenomenon* [9] is another property that is often observed for high-dimensional data. For the hubness phenomenon, one counts for each point of a data set how often this points occurs among the $k$ nearest neighbours of other points where $k$ is a fixed constant. Especially for uniformly distributed data one would expected that each point roughly occurs equally often among the $k$ nearest neighbours of other points. However, for higher dimensions this is not true. This means that some points – called hubs – occur extremely often among the $k$ nearest neighbours of other points.

Low et al. [10] argue that the hubness phenomenon is actually not a direct effect of high-dimensional data, but a boundary effect. The reason why it is connected to high-dimensional data is simply that for higher dimensions the proportion of the data at the boundary of the data set increases exponentially with the dimensions.

Before we discuss what these phenomena mean for cluster analysis, we give a brief overview on basic clustering techniques in the following section.

## 3 Cluster analysis

Cluster analysis aims at grouping a data set into clusters where data objects within a cluster are similar to each other and different from data objects in other clusters. Especially for data with real-valued attributes, similarity is usually defined based on a notion of distance like the Euclidean or the Manhattan distance. In order to understand how the CoN and the hubness phenomenon might affect cluster analysis, it is important to understand how these distances are used in the clustering process. Therefore, we provide a brief overview on cluster analysis which is far from being complete. For more details on cluster analysis we refer to books like [11, 1].

Hierarchical agglomerative clustering is a relational clustering technique that is based on a distance matrix in which the pairwise distances of the data objects are

entered. For data with real-valued attributes, these distances are often simply the Euclidean distances between the data points. Initially, each data point is considered as a separate cluster and then – step by step – the closest points or clusters are joint together to form a larger cluster until all data points end up in one cluster. The number of clusters is determined based on the dendrogram that shows in which order the clusters were joint together and how large the distances were between the joint clusters.

Hierarchical clustering cannot avoid at least quadratic complexity in the number of data points, since the required distance matrix is already quadratic in the number of data objects. Therefore, hierarchical clustering can be problematic for larger data sets. And when we deal with high-dimensional data, we should expect the data set not to be too small, at least compared to the number of dimensions.

Prototype based clustering like the well known k-means algorithm [12] tries to avoid this complexity problem. From a purely algorithmic point of view, k-means clustering can be described as follows. First the number of clusters $k$ must be fixed. Then each of the $k$ clusters is represented by a prototype $v_i \in \mathbb{R}^m$ when we want to cluster $m$-dimensional data. These prototypes are chosen randomly in the beginning. Then each data vector is assigned to the nearest prototype (with respect to the Euclidean distance). Then each prototype is replaced by the centre of gravity of those data assigned to it. The alternating assignment of data to the nearest prototype and the update of the prototypes as cluster centres is repeated until the algorithm converges, i.e., no more changes happen.

This algorithm can also be seen as a strategy for minimising the objective function

$$ f \;=\; \sum_{i=1}^{k} \sum_{j=1}^{n} u_{ij} d_{ij} \tag{2} $$

under the constraints

$$ \sum_{i=1}^{k} u_{ij} \;=\; 1 \qquad \text{for all } j = 1, \ldots, n \tag{3} $$

where $u_{ij} \in \{0, 1\}$ indicates whether data vector $x_j$ is assigned to cluster $i$ ($u_{ij} = 1$) or not ($u_{ij} = 0$). $d_{ij} = \parallel x_j - v_i \parallel^2$ is the squared Euclidean distance between data vector $x_j$ and cluster prototype $v_i$.

One of the problems of k-means clustering is that it can be quite sensitive to the choice of the initial prototypes and can easily get stuck in local minima of the objective function (2), i.e. leading to counterintuitive clustering results.

Fuzzy k-means clustering[4] [13, 14] is a generalisation of k-means clustering and is less sensitive to the initialisation, since the number of local minima of the objective function can be reduced [15]. For fuzzy clustering, the constraints $u_{ij} \in \{0, 1\}$ is relaxed to $u_{ij} \in [0, 1]$. However, this change alone would still yield the same optimum

---

[4] For fuzzy clustering, the number of clusters is usually denoted by $c$, so that fuzzy c-means clustering (FCM) is the common term. But for consistency reasons, we always denote the number of clusters by $k$.

of the objective function (2). Therefore, a so-called fuzzifier $w > 1$ is introduced in the objective function:

$$f = \sum_{i=1}^{k} \sum_{j=1}^{n} u_{ij}^{w} d_{ij}. \tag{4}$$

k-means and its fuzzified version can also be extended to adapt to other than spherical cluster shapes [16] or clusters of different sizes [17]. For a more detailed overview on fuzzy cluster analysis we refer to [18, 19].

Gaussian mixture models can be seen as a probabilistic model for clustering where the data from each cluster are assumed to represent a sample from a multidimensional normal distribution. The expectation maximisation algorithm (EM) can be viewed as a generalisation of the k-means algorithm where prototypes in the form the parameters of the normal distributions and assignments to clusters in terms of probabilities are estimated in an alternating fashion.

All the above mentioned clustering algorithm assume that the number of clusters is known. There are, of course, methods to estimate the number of clusters based on cluster validity measures or techniques from model selection like the Bayes information criterion. But there are also clustering techniques that automatically determine the number of clusters. These clustering algorithms are usually density-based, i.e. they try to find regions of higher data concentration that form clusters. Examples for such algorithms are DBSCAN [20], DENCLUE [21] and OPTICS [22].

Subspace clustering – for an overview see for instance [23] – refers to approaches that are especially tailored for high-dimensional data. Their main idea is not to cluster the data in all dimensions, but to look for clusters in projections to lower dimensions. These projections can be axes-parallel, but do not have to be. Essentially, subspace clustering comprises two component that are combined in the algorithm: Finding a suitable projection and applying a clustering algorithm in the corresponding subspace defined by the projection.

Before we take a closer look at high-dimensional data in connection with cluster analysis, we mention a few examples where it is of interest to cluster high-dimensional data. First of all, when can we speak of high-dimensional data? The CoN phenomenon and the hubness effect can already be quite noticeable 30 dimensions. Data of this dimensionality can easily be found in many applications like industrial production where simultaneously measurements from a larger number of sensors is recorded constantly. Patient data in medicine including the laboratory results can also have a significant number of attributes.

But there are also applications having 10,000 or more dimensions. High throughput technologies like microarrays can measure the expression of far more than 10,000 genes easily. Often, the genes are clustered, which would mean that the number of data objects exceeds 10,000, but not the number of dimensions. But sometimes it is also interesting to cluster microarray data from different experiments or different individuals which leads to such high-dimensional data, since then each gene or its expression value is considered as an attribute (see for instance [24]). Other high throughput technologies like mass spectrometry for proteomics and in combination with gas chromatography for metabolomics or next generation sequencing also produce data with thousands of

dimensions. In [25] growth curves of more than 4000 mutants under more than 100 conditions were measured by the VITEK®2 technology and clustered.

So there is an obvious need to cluster data with more than 30 up to tens of thousands dimensions. But what are clusters in such high dimensions? How do they look like and what can we expect from them? The following section will discuss these question in more detail.
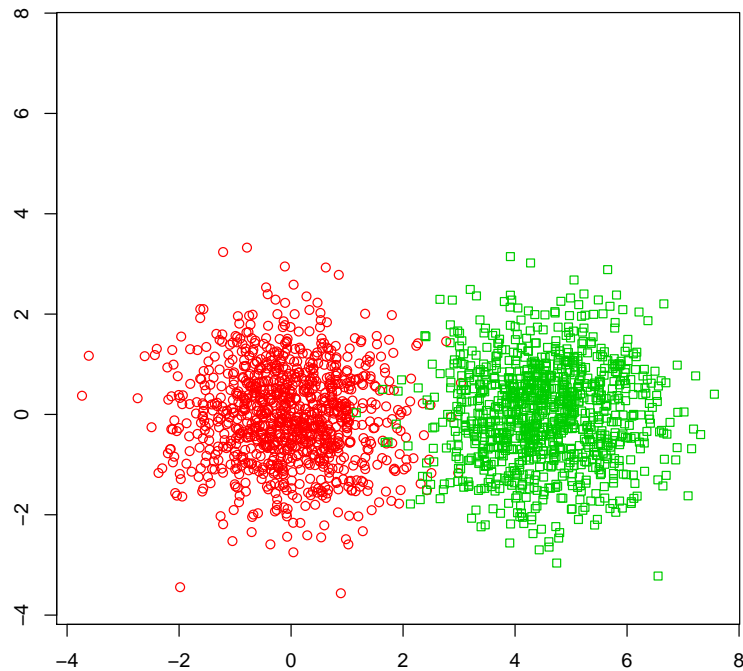


**Fig. 1.** An artificial data set with two clusters characterised by the $x$-values.

## 4   What are clusters, especially in higher dimensions?

The CoN phenomenon affects high-dimensional data in general. Especially, nearest neighbour search and database queries of the top $k$ similar cases might not be meaningful anymore for high-dimensional data [26].

But what are the consequences for cluster analysis which is usually also based on distance notions? In order to discuss these consequences, it is essential to first clarify what we mean by or what we expect from a cluster in high dimensions. However, to specify what we expect from a high-dimensional cluster is not so obvious as the following two artificial examples show. Both examples contain two clusters, each of them having 1000 data objects.

The first example is essentially a one-dimensional example. Both clusters are generated by normal distributions with variance 1. But the first cluster has expected value 0, whereas the second cluster has expected value 4.5. In order to make it a high-dimensional example, we add additional dimensions that do not contribute to the distinction of the clusters. For both clusters, the values of the additional dimensions come from standard normal distributions, i.e. with expected value 0 and variance 1.

Figure 1 shows the two-dimensional data set. The relevant attribute to distinguish the two clusters is shown on the $x$-axis, whereas the first irrelevant attribute is shown on the $y$-axis.
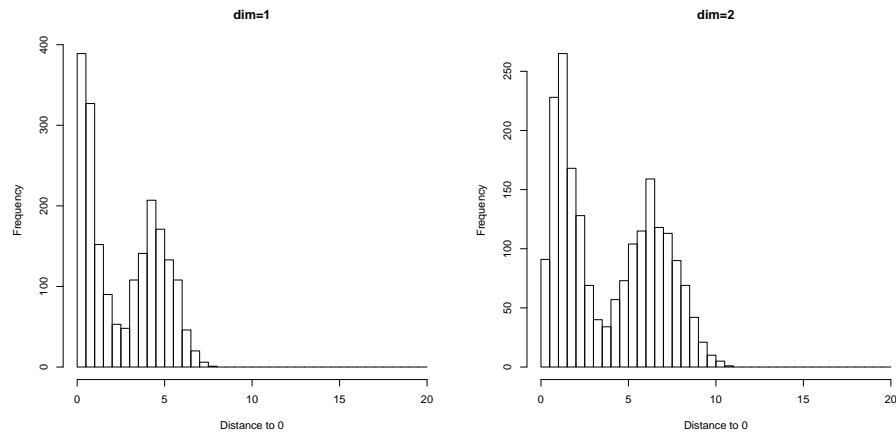


**Fig. 2.** Distribution of the distances of the data to the origin of the coordinate system in the first example for one (left) and two (right) dimensions.

We now consider the distribution of the Euclidean distances of the points in the data set to the origin of the coordinate system. Figure 2 shows this distribution for the one-dimensional case on the left hand side when only the relevant attribute is considered. The histogram has one peak for each cluster. The first cluster is represented by the peak at 1. This corresponds to the data from the standard normal distribution scattering around zero with an average distance of 1, since the variance is 1. The second peak is at 4.5, the expected value of the second normal distribution.

The right-hand side of Figure 2 shows the distribution of the distances in the two-dimensional case, i.e. when in addition to the relevant attribute, one irrelevant attribiute is present that does not contribute to the distinction of the clusters.
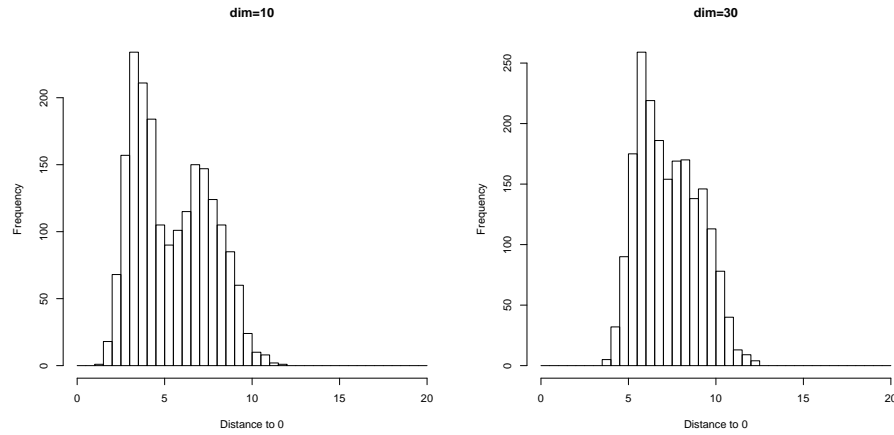


**Fig. 3.** Distribution of the distances of the data to the origin of the coordinate system in the first example for 10 (left) and 30 (right) dimensions.

When we increase the number of attributes whose distributions are the same for both clusters, the two peaks in the histograms in Figure 2 start to melt together, as can be seen in Figure 3 which shows the histograms for 10 and 30 dimensions. For 10 dimensions, two peaks can still be identified, whereas they are almost invisible in 30 dimensions.

For 50 dimensions in Figure 4 on the left-hand side, only one peak can be identified although the distribution is still skewed. For 200 dimensions in Figure 4 on the right-hand side even the skewness has vanished.

Table 1 shows some statistical characteristics of the intra- and the inter-cluster distances for this example. The intra-cluster distances are obtained by computing the Euclidean distance between each pair of points from the the same cluster. For the inter-cluster distances, the two points are always chosen from different clusters. For low dimensions, the distributions of the intra- and the inter-cluster distances differ significantly. But with an increasing number of dimensions, these two distributions become more and more similar.

What does this simple artificial example tell us about clusters in high dimensions? There is no doubt about the two clusters in low dimensions as Figure 1 illustrates it clearly for two dimensions and as the histograms indicate it with their two peaks in low dimensions. But are these two clusters still present in high-dimensions? Neither the distance histograms for higher dimensions nor the vanishing difference between the intra- and the inter-cluster distance distributions indicate that there are two clusters present.
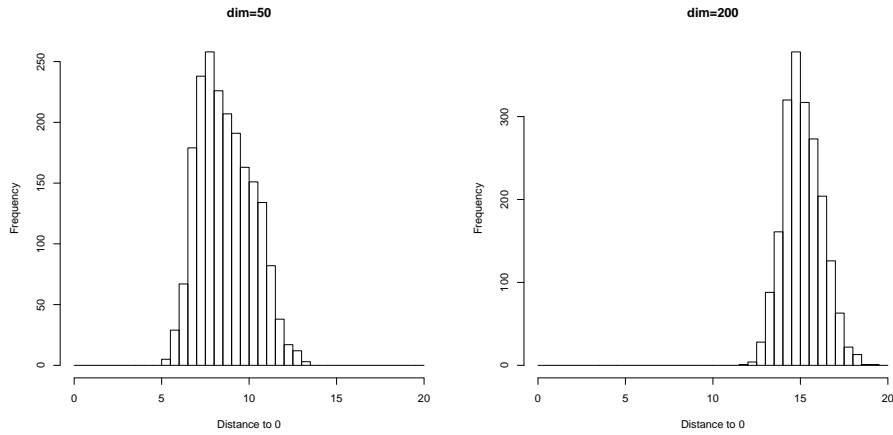
**Fig. 4.** Distribution of the distances of the data to the origin of the coordinate system in the first example for 50 (left) and 200 (right) dimensions.

Before we further discuss this example, we consider another artificial example. Again, there are two clusters. In contrast to the first example, there is not a single attribute by which the two clusters can be distinguished, but each attribute contributes a little bit to the distinction of the two clusters. For each attribute, we assume a normal distribution with variance 1 for both clusters, but with expected value 0 for the first, and expected value 1 for the second cluster. Again, each cluster contains 1000 data objects.

Figure 5 shows the data for two dimensions. The two clusters are visually indistinguishable. Only the use of different symbols for the two clusters provides visual information about the presence of two clusters.

In the same fashion as for the first example, Figures 6–8 show the distributions of the distances to the origin of the coordinate system for the second example for different dimensions. We can observe exactly the opposite effect as in the first example. For low dimensions, there is only one peak and the two clusters cannot be detected by looking at the histograms. For about 30 dimensions, the single peak starts to be separated into two peaks and the separation of the two peaks is almost perfect for 200 dimensions.

Table 2 contains the characteristics of the inter-cluster distances for the second example. The intra-cluster distances have, of course, the same characteristics as in the first example, i.e. the values correspond to the ones in Table 1 (the non-bold face rows). It is no surprise that we can observe the opposite effect as in the first example. For low dimensions, the distributions of the intra- and the inter-cluster distances are quite similar, but with an increasing number of dimensions, they become more and more distinguishable.

| Dimensions | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| intra/**inter** | 1 | 2 | 3 | 5 | 10 | 20 | 30 | 50 | 100 | 200 |
| Minimum | < 0.001 | 0.001 | 0.026 | 0.237 | 0.871 | 2.479 | 3.782 | 6.031 | 9.841 | 15.967 |
| **Minimum** | < 0.001 | 0.048 | 0.129 | 0.384 | 1.733 | 3.318 | 4.732 | 6.336 | 10.824 | 16.613 |
| 5% quantile | 0.088 | 0.537 | 1.053 | 1.860 | 3.269 | 5.136 | 6.649 | 8.882 | 12.954 | 18.771 |
| **5% quantile** | 2.218 | 3.377 | 3.782 | 4.343 | 5.390 | 6.925 | 8.193 | 10.192 | 13.995 | 19.560 |
| 25% quantile | 0.450 | 1.283 | 1.969 | 2.877 | 4.309 | 6.167 | 7.660 | 9.884 | 13.932 | 19.752 |
| **25% quantile** | 3.607 | 5.278 | 5.597 | 6.064 | 6.962 | 8.334 | 9.515 | 11.431 | 15.110 | 20.626 |
| Median | 0.954 | 2.004 | 2.772 | 3.699 | 5.107 | 6.931 | 8.394 | 10.602 | 14.624 | 20.446 |
| **Median** | 4.561 | 6.600 | 6.883 | 7.306 | 8.107 | 9.379 | 10.496 | 12.340 | 15.925 | 21.390 |
| Mean | 1.132 | 2.161 | 2.895 | 3.788 | 5.169 | 6.972 | 8.422 | 10.625 | 14.635 | 20.452 |
| **Mean** | 4.549 | 6.595 | 6.892 | 7.324 | 8.144 | 9.426 | 10.544 | 12.379 | 15.959 | 21.405 |
| 75% quantile | 1.633 | 2.869 | 3.687 | 4.602 | 5.965 | 7.733 | 9.152 | 11.340 | 15.329 | 21.143 |
| **75% quantile** | 5.501 | 7.913 | 8.176 | 8.558 | 9.285 | 10.467 | 11.521 | 13.285 | 16.773 | 22.167 |
| 95% quantile | 2.788 | 4.314 | 5.157 | 6.022 | 7.281 | 8.943 | 10.295 | 12.439 | 16.349 | 22.158 |
| **95% quantile** | 6.844 | 9.795 | 10.036 | 10.383 | 11.025 | 12.091 | 13.061 | 14.696 | 18.041 | 23.306 |
| Maximum | 6.990 | 9.952 | 10.046 | 10.490 | 11.683 | 12.914 | 13.942 | 15.943 | 19.802 | 25.211 |
| **Maximum** | 10.420 | 14.741 | 14.753 | 15.173 | 16.061 | 17.083 | 17.452 | 18.608 | 22.115 | 27.236 |

**Table 1.** Statistical characteristics of the intra- and inter-cluster distances for the first example.

## 5   Consequences for clustering algorithms

The two examples in the previous section illustrate, how the CoN phenomenon influences cluster analysis in high dimensions. If we assume that the clusters are defined by only a few variables and most of the other variables have no connection to the clusters at all, then the CoN phenomenon destroys the distinguishability of clusters as demonstrated by the first example. But when (almost) all attributes contribute at least a little bit to the distinction of the clusters, the CoN phenomenon even helps to identify the clusters.

How many "noise" dimensions can be tolerated in the first example, until we say that there are no longer two clusters? How many dimensions do we need in the second example, until we can really speak of two clusters?

Of course, for real world data, the situation is usually more complicated. There might be some attributes that lead to good clusters and other attributes which are simply "noise" in terms of the clustering. And in contrast to the two examples, attributes might also be correlated. The first example has shown that a limited number of "noise" attributes or "noise" dimensions can be tolerated. But what happens when there are different combinations of attributes, leading to different clustering results? Which clusters should we trust when there are completely different clusters for different combinations of attributes, i.e. in different subspaces?

There is, of course, a clear argument for some kind of subspace clustering approach. Typically, the number of clusters we expect or we are searching for in a data set, is limited. In many applications, even 20 clusters is already a very large number. When
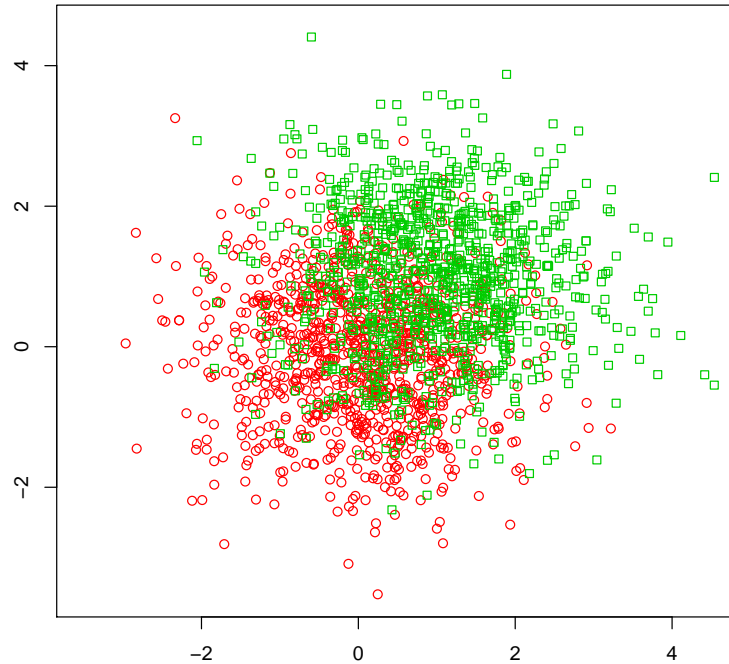
**Fig. 5.** An artificial data set with two clusters that are difficult to distinguish in low dimensions.

we have $k$ clusters and think in terms of prototype-based clustering, the cluster centres lie in an at most $(k-1)$-dimensional hyperplane. So this means that, at least when the clustering result is good, the clustering could have been carried out on the projection of the data to this $(k-1)$-dimensional hyperplane.

The idea of finding interesting patterns in lower dimensions in high-dimensional data sets by suitable projections is well known in data analysis as a technique called projection pursuit [27]. However, projection pursuit focuses mainly on two- and three-dimensional projections for visualisation purposes and does not specifically aim at finding clusters. Since projections of high-dimensional data to low dimensions tend to resemble normal distributions, one way to carry out projection pursuit is to generate random two- or three-dimensional projections and to apply a test for normality. Those projections for which the normality hypothesis is rejected are then presented to the user for further inspection or the projections can be ranked according to their p-values that the tests for normality yielded. However, a deviation from a normal distribution does not necessarily indicate that there are clusters in the data set.
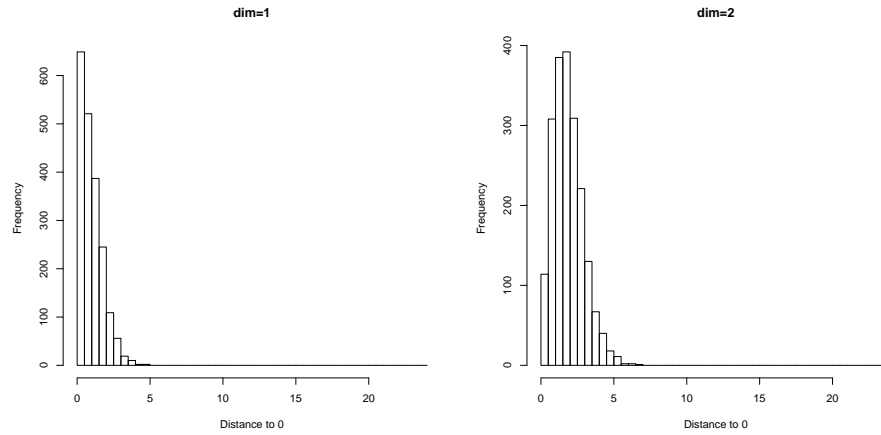
**Fig. 6.** Distribution of the distances of the data to the origin of the coordinate system in the second example for one (left) and two (right) dimensions.

Finding the right projection is a crucial problem in subspace clustering. It can be seen as a feature selection strategy [28] when only axes-parallel projections are considered.

It should be taken into account that we have to deal with the problem of multiple testing for very high-dimensional data like in the case of microarrays: multiple testing in the sense that we test a larger number of projection for the presence or absence of clusters. When we have many attributes, the number of possible projections is extremely large – or even infinite if we drop the restriction to axes-parallel projections. This means that the probability that a certain projection contains clusters just by chance might not be negligible.

| Dimensions | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | 1 | 2 | 3 | 5 | 10 | 20 | 30 | 50 | 100 | 200 |
| **Minimum** | < 0.001 | 0.004 | 0.026 | 0.210 | 1.117 | 2.656 | 4.680 | 7.195 | 12.355 | 19.791 |
| **5% quantile** | 0.114 | 0.687 | 1.340 | 2.382 | 4.082 | 6.385 | 8.212 | 10.984 | 15.998 | 23.112 |
| **25% quantile** | 0.577 | 1.623 | 2.481 | 3.611 | 5.327 | 7.612 | 9.428 | 12.168 | 17.165 | 24.243 |
| **Median** | 1.208 | 2.513 | 3.467 | 4.588 | 6.267 | 8.517 | 10.301 | 13.016 | 17.988 | 25.037 |
| **Mean** | 1.398 | 2.672 | 3.587 | 4.679 | 6.325 | 8.561 | 10.330 | 13.039 | 17.998 | 25.044 |
| **75% quantile** | 2.025 | 3.548 | 4.560 | 5.648 | 7.258 | 9.462 | 11.203 | 13.881 | 18.819 | 25.836 |
| **95% quantile** | 3.341 | 5.206 | 6.260 | 7.288 | 8.765 | 10.882 | 12.540 | 15.178 | 20.031 | 27.001 |
| **Maximum** | 7.284 | 10.488 | 10.747 | 13.522 | 14.648 | 16.883 | 17.821 | 19.528 | 24.085 | 31.059 |

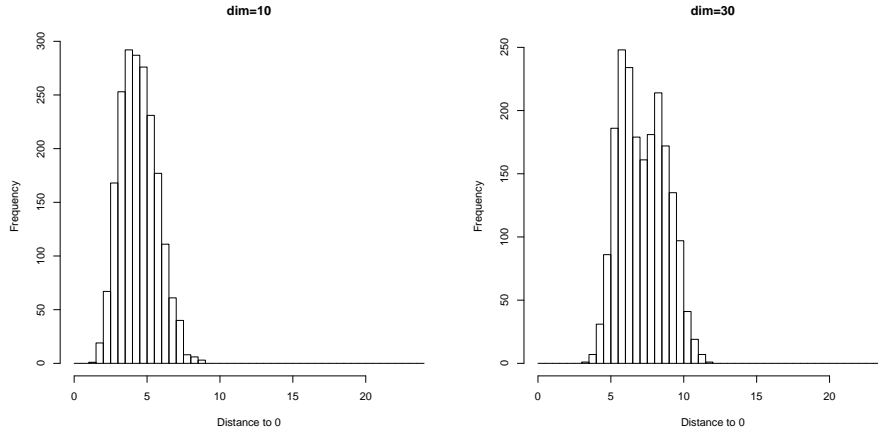**Table 2.** Statistical characteristics of the inter-cluster distances for the second example.

**Fig. 7.** Distribution of the distances of the data to the origin of the coordinate system in the second example for 10 (left) and 30 (right) dimensions.

As an example, consider a data set with 200 data objects and 10,000 attributes which are all independent samples from a uniform distribution on the unit interval. So we have 200 uniformly distributed data points in the unit hypercube of dimension 10,000. There are, of course, no clusters. But we do not know this fact, since in reality we would not know from which distribution the data were generated. In order to find clusters in subspaces, we only consider projections to three dimensions here. Figure 9 illustrates a projection where we can see two clusters. One cluster is above the diagonal plane of the unit cube, the other one below.

What is the chance that we can find such a projection for our uniformly distributed data from the 10,000-dimensional unit hypercube? Assume the separation between the clusters should be at least 0.1 units, i.e. the plane in Figure 9 would be turned into a box of height[5] 0.1. What is the probability that no projection will look like this, i.e. that each projection contains at least one point within the separating box? The box has a volume of

$$\sqrt{1^2 + \left(\frac{1}{2}\right)^2} \cdot \sqrt{1^2 + \left(\frac{1}{2}\right)^2} \cdot 0.1 \; = \; 0.125,$$

so that the remaining volume of the cube is $1 - 0.125 = 0.875$. The probability that all points lie in this remaining cube outside the separating box, i.e. that we have two artificial clusters, is $0.875^{200}$ and the probability that these artificial random cluster do not occur for a single projection to three dimensions is $1 - 0.875^{200}$. The probability that we will not find such random clusters in any of the $10000 \cdot 9999 \cdot 9998$ possible projections to three dimensions is

$$\left(1 - 0.875^{200}\right)^{10000 \cdot 9999 \cdot 9998} \; \approx \; 0.08.$$

---

[5] In Figure 9 the separation between the two clusters is chosen larger for illustration purposes.
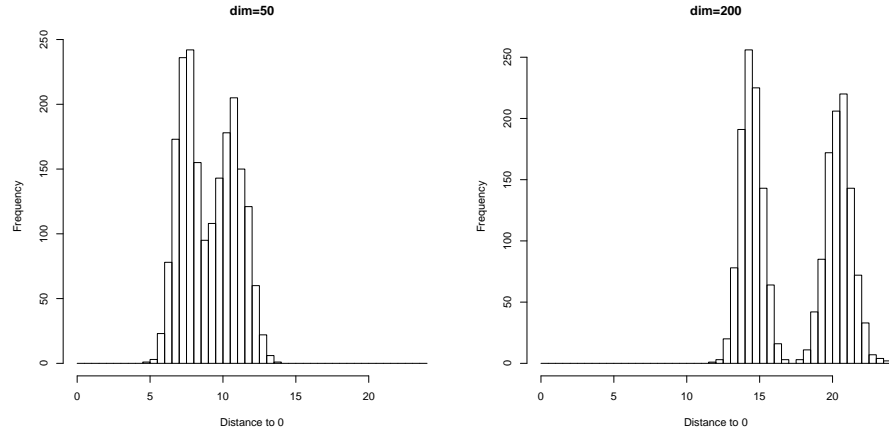
**Fig. 8.** Distribution of the distances of the data to the origin of the coordinate system in the second example for 50 (left) and 200 (right) dimensions.

So there is only an 8% chance that we will not find these spurious random clusters in any of the three-dimensional projections of our data set from a high-dimensional uniform distribution. It should be noted that we have only considered projections to three dimensions and a specific separating plane between the clusters. If we allow projections to more than three dimensions and take into account that the random clusters might be separated by other planes or geometric shapes, the situation gets much worse and the probability that we find random clusters in a projection is almost 1, even if we have more than 200 data points.

Therefore, it is highly recommended to carry out some additional tests to verify whether clusters found by subspace clustering in very high-dimensional data set are not just random effects. One can apply a permutation test in which the values in each column of the data table are randomly permuted. Then the clustering algorithm is applied to the randomly permuted data. When the clustering algorithm still finds clusters in this permuted data set, one cannot trust the clusters that were found in the original data set. The permutation test should be carried out more than once.

A Monte Carlo test is also possible by generating random data of the same dimension and with the same number of data objects, but using a distribution where no clusters should be found like the uniform distribution in the unit hypercube that we have considered above. When the algorithm can find clusters in such a random data set, the clusters in the real data set might again be just random artifacts.

Subspace clustering is definitely needed when we assume that our data set is more like the first example described in Section 4, i.e. there are a few attributes that contribute to the clusters and the large majority of attributes is just "noise". In this case, clustering algorithms taking all attributes into account, would have little or no chance to discover the clusters. The CoN and the hubness phenomenon will hide the clusters.
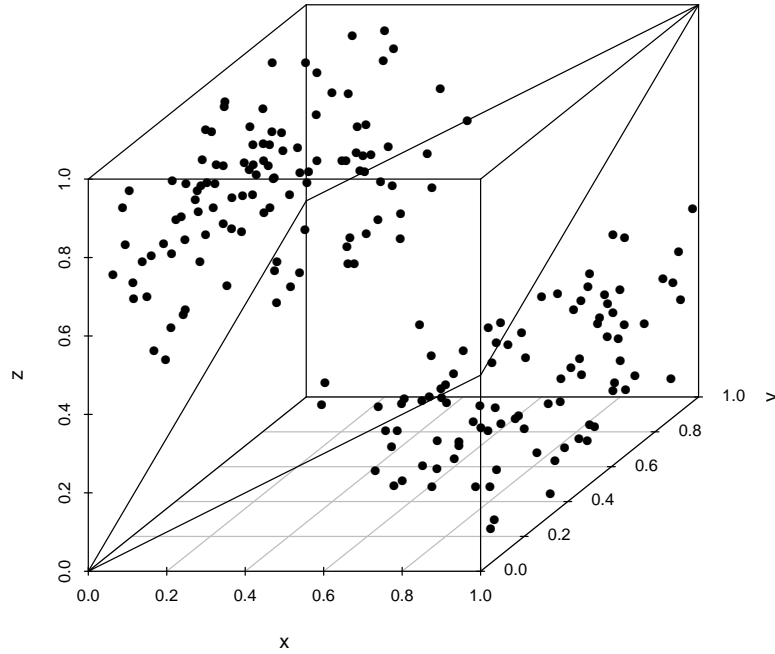
**Fig. 9.** A possible projection which contains clusters only by chance.

But how is the situation when the data set is more of the type as the second example described in Section 4, i.e. most of the attributes contribute a little bit to the clusters and only a few "noise" attributes might be present? Figure 8 indicates that the CoN phenomenon can even make it easier to find the clusters. The reason is that the CoN phenomenon is not applicable to the whole data set. The assumption that the relative variance goes to zero is not satisfied here. The relative variance in this simple example is strictly positive due to the two clusters whose centres have a distance of $\sqrt{m}$. The expected distance to the origin is $\frac{\sqrt{m}}{2}$. So the relative variance is roughly $\frac{1}{4}$. The CoN phenomenon does occur, but in each cluster separately as can be seen from Figure 8 where the two peaks in the histograms for the clusters become not only better separated, but also quite narrow for high dimensions.

However, the situation is not as positive and simple as it seems. When we look at prototype-based clustering, it is not a problem of the objective function. For an ideal data set as the second example in Section 4, the objective function will have a clear global minimum at the centre of the two clusters. But when we have more than just two

clusters, it becomes a problem of local minima of the objective function and therefore a problem of a good initialisation. Since the location of the cluster centres is not known a priori, the initial prototypes are placed "somewhere" and then suffer from the CoN phenomenon on the level of the clusters in the sense that all clusters have roughly the distance to them unless a prototype is located close to a true cluster centre.
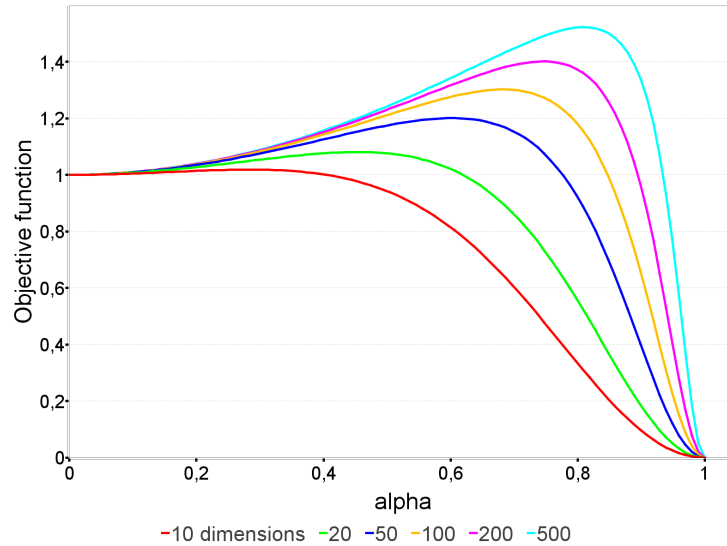


**Fig. 10.** The objective function of fuzzy clustering has a local minimum in the centre of all data points for high-dimensional data.

In [29] it was demonstrated that k-means clustering has difficulties to find clusters in high dimensions, even when the clusters are well separated. One might suspect that this is the well-known sensitivity of k-means clustering to the initialisation and that fuzzy clustering might yield better results. But the contrary is the case. For fuzzy clustering, all or most of the prototypes tend to converge in the centre of gravity of the whole data set.

What is the reason for this surprising result? Figure 10 from [29] explains this effect. It shows the objective function (4) of fuzzy clustering reduced to one parameter for a specific data set. The data set consists of a fixed number of well-separated clusters – each of them concentrated in a single point – distributed uniformly on the surface of an $(m - 1)$-dimensional unit hypersphere. The cluster prototypes are first all placed into the origin, i.e. the centre of gravity of all the data points. Then the cluster prototypes are moved along the lines connecting each cluster prototype with one of the true cluster centres. So at 0 on the $x$-axis in Figure 10 all prototypes are at the origin (radius=0), at 0.5 they are halfway between the origin and the true cluster centres and at 1 each of the prototypes is placed exactly in one of the cluster centres. As can be seen from the figure,

the clear global minimum of the objective function is at 1, i.e. when all prototypes are placed in the true cluster centres. But there is a local minimum at the origin, separated by a local maximum from the global minimum. The local maximum is shifted more to the right for higher dimensions. Since the algorithm to minimise the objective function of fuzzy clustering can be view as a gradient descent technique [30], the cluster prototypes will end up in the local minimum at the origin when the initialisation is not close enough to the true cluster centres.

These considerations about the objective function for fuzzy clustering in high dimensions demonstrate that the CoN phenomenon occurs on the level of the cluster centres in the sense that all cluster centres have roughly the same distance from a prototype that is not placed close to a true cluster centre. So this is not a problem of the objective function which clearly shows a global minimum at the correct cluster centres. It is a problem of the optimisation algorithm. In contrast to this example where the clusters are well separated, the first example in Section 4 would also cause a problem for the objective function, since the minimum would not be pronounced very clearly.

There are ways to partly avoid these problems. One way is to try to adjust the fuzzifier $w$ in the objective function (4) depending on the number of dimensions. The higher the number of dimensions, the smaller, but of course larger than 1, the fuzzifier should be chosen. A better way to avoid the tedious adjustment of the fuzzifier is to use a polynomial fuzzifier function [31] that replaces the power function $u^w$ by a quadratic polynomial of the form $w \cdot u^2 + (1 - w) \cdot u$ with $u \in [0, 1]$. This leads to a convex combination of the standard k-means clustering objective function (2) and the objective function for fuzzy clustering (4) with fuzzifier 2.

Density-based clustering suffers from the problem that density can extremely vary in high dimensions and it is very difficult to adjust the parameter settings.

## 6   Conclusions

Clustering high-dimensional data is a difficult task. The problem starts already with the understanding of how a cluster should be characterised. Will only a few dimensions or attributes contribute to the clustering and the large majority of attributes is considered as "noise"?

We have seen with the first example in Section 4 that a limited number of "noise" attributes can be tolerated without destroying the clusters. If the data set is expected to contain too many "noise" attributes that are irrelevant for the clustering, subspace clustering techniques are needed. Subspace clustering drastically increases the complexity of clustering, since not only clusters need to be found but also the right subspace. Apart from this, for very high-dimensional data, subspace clustering might lead to finding spurious clusters that "look good", but are just random effects as we have illustrated in Section 5. Therefore, validation techniques like permutation or Monte Carlo tests should be applied to get an idea for the chances of finding spurious clusters.

If the number of "noise" attributes is limited, subspace clustering might not be necessary. However, clustering algorithms might still suffer from the CoN phenomenon. As we have seen, if the clusters are well separated, this is not a problem of the cluster model – the objective function – but a problem of the algorithm to find the best cluster

model or to minimise the objective function. As for subspace clustering, it is highly recommended to evaluate the clusters. However, here other methods than permutation or Monte Carlo tests might be better suited like cross-validation in the sense of resampling [32] or the application of other cluster validity measures. A good overview on cluster validity measures can be found in [33]. These techniques are also often used to determine the number of clusters.

Missing values are a problem that is usually not considered in cluster analysis [34]. But missing values will occur with larger probability in high-dimensional data. This is still an open problem.

# References

1. Berthold, M., Borgelt, C., Höppner, F., Klawonn, F.: Guide to Intelligent Data Analysis: How to Intelligently Make Sense of Real Data. Springer, London (2010)
2. Bellmann, R.: Adaptive Control Processes: A Guided Tour. Princeton Univ. Press (1961)
3. Beyer, K.S., Goldstein, J., Ramakrishnan, R., Shaft, U.: When is "nearest neighbor" meaningful? In Beeri, C., Buneman, P., eds.: ICDT. Volume 1540 of Lecture Notes in Computer Science., Springer (1999) 217–235
4. Durrant, R.J., Kabán, A.: When is 'nearest neighbour' meaningful: A converse theorem and implications. J. Complexity **25**(4) (2009) 385–397
5. François, D., Wertz, V., Verleysen, M.: The concentration of fractional distances. IEEE Trans. Knowl. Data Eng. **19**(7) (2007) 873–886
6. Aggarwal, C.C.: Re-designing distance functions and distance-based applications for high dimensional data. SIGMOD Record **30**(1) (2001) 13–18
7. Hsu, C.M., Chen, M.S.: On the design and applicability of distance functions in high-dimensional data space. IEEE Trans. Knowl. Data Eng. **21**(4) (2009) 523–536
8. Jayaram, B., Klawonn, F.: Can unbounded distance measures mitigate the curse of dimensionality? Data Mining, Modelling and Management **4** (2012) 361–383
9. Radovanović, M., Nanopoulus, A., Ivanović, M.: Hubs in space: Popular nearest neighbors in high-dimensional data. Machine Learning Research **11** (2010) 2487–2531
10. Low, T., Borgelt, C., Stober, S., Nürnbberger, A.: The hubness phenomenon: Fact or artifact? In Borgelt, C., Ángeles Gil, M., Sousa, J., Verleysen, M., eds.: Towards Advanced Data Analysis by Combining Soft Computing and Statistics. Springer, Berlin (2013) 267–278
11. Evertt, B., Landau, S.: Cluster Analysis. 5th edn. Wiley, Chichester (2011)
12. Duda, R., Hart, P.: Pattern Classification and Scene Analysis. Wiley, New York (1973)
13. Dunn, J.: A fuzzy relative of the isodata process and its use in detecting compact well-separated clusters. Cybernetics and Systems **3**(3) (1973) 32–57
14. Bezdek, J.: Pattern Recognition with Fuzzy Objective Function Algorithms. Plenum Press, New York (1981)
15. Jayaram, B., Klawonn, F.: Can fuzzy clustering avoid local minima and undesired partitions? In Moewes, C., Nürnberger, A., eds.: Computational Intelligence in Intelligent Data Analysis. Springer, Berlin (2012) 31–44
16. Gustafson, D., Kessel, W.: Fuzzy clustering with a fuzzy covariance matrix. In: IEEE CDC, San Diego (1979) 761–766
17. Keller, A., Klawonn, F.: Adaptation of cluster sizes in objective function based fuzzy clustering. In Leondes, C., ed.: Intelligent Systems: Technology and Applications vol. IV: Database and Learning Systems. CRC Press, Boca Raton (2003)
18. Bezdek, J., Keller, J., Krishnapuram, R., Pal, N.: Fuzzy Models and Algorithms for Pattern Recognition and Image Processing. Kluwer, Boston (1999)

19. Höppner, F., Klawonn, F., Kruse, R., Runkler, T.: Fuzzy Cluster Analysis. Wiley, Chichester (1999)
20. Ester, M., Kriegel, H.P., Sander, J., Xu, X.: A density-based algorithm for discovering clusters in large spatial databases with noise, AAAI Press (1996) 226–231
21. Hinneburg, A., Gabriel, H.H.: Denclue 2.0: Fast clustering based on kernel density estimation. In: Proceedings of the 7th International Symposium on Intelligent Data Analysis. (2007) 70–80
22. Ankerst, M., Breunig, M., Kriegel, H.P., Sander, J.: Optics: Ordering points to identify the clustering structure, ACM Press (1999) 49–60
23. Kriegel, H.P., Kröger, P., Zimek, A.: Clustering high-dimensional data: A survey on subspace clustering, pattern-based clustering, and correlation clustering. ACM Trans. Knowl. Discov. Data **3**(1) (March 2009) 1–58
24. Kerr, G., Ruskin, H., Crane, M.: Techniques for clustering gene expression data. Computers in Biology and Medicine **38**(3) (2008) 383–393
25. Pommerenke, C., Müsken, M., Becker, T., Dötsch, A., Klawonn, F., Häussler, S.: Global genotype-phenotype correlations in pseudomonas aeruginosa. PLoS Pathogenes **6**(8) (2010) 10.1371/journal.ppat.1001074
26. Hinneburg, A., Aggarwal, C., Keim, D.: What is the nearest neighbor in high dimensional spaces? In Abbadi, A.E., Brodie, M.L., Chakravarthy, S., Dayal, U., Kamel, N., Schlageter, G., Whang, K.Y., eds.: VLDB, Morgan Kaufmann (2000) 506–515
27. Cook, D., Buja, A., Cabrera, J.: Projection pursuit indices based on orthonormal function expansion. Journal of Computational and Graphical Statistics **2** (1993) 225–250
28. Wang, S., Zhu, J.: Variable selection for model-based high-dimensional clustering and its application to microarray data. Biometrics **64** (2008) 440–448
29. Winkler, R., Klawonn, F., Kruse, R.: Fuzzy c-means in high dimensional spaces. Fuzzy System Applications **1** (2011) 1–17
30. Höppner, F., Klawonn, F.: A contribution to convergence theory of fuzzy c-means and its derivatives. IEEE Transactions on Fuzzy Systems **11** (2003) 682–694
31. Klawonn, F., Höppner, F.: What is fuzzy about fuzzy clustering? understanding and improving the concept of the fuzzifier. In Berthold, M.R., Lenz, H.J., Bradley, E., Kruse, R., Borgelt, C., eds.: Advances in Intelligent Data Analysis. Volume V., Berlin, Springer (2003) 254–264
32. Borgelt, C.: Resampling for fuzzy clustering. Int. Journal of Uncertainty, Fuzziness and Knowledge-based Systems **15** (2007) 595–614
33. Borgelt, C.: Prototype-based Classification and Clustering. Habilitation thesis (2006)
34. Himmelspach, L., Conrad, S.: Clustering approaches for data with missing values: Comparison and evaluation. In: ICDIM'10. (2010) 19–28