Outlier-robust Clustering using Independent Components

Christian Böhm
University of Munich
Munich, Germany
boehm@dbs.ifi.lmu.de

Christos Faloutsos
Carnegie Mellon University
Pittsburgh, PA, USA
christos@cs.cmu.edu

Claudia Plant
Technical University of Munich
Munich, Germany
plant@lrz.tum.de

ABSTRACT

How can we efficiently find a clustering, i.e. a concise description of the cluster structure, of a given data set which contains an unknown number of clusters of different shape and distribution and is contaminated by noise? Most existing clustering methods are restricted to the Gaussian cluster model and are very sensitive to noise. If the cluster content follows a non-Gaussian distribution and/or the data set contains a few outliers belonging to no cluster, then the computed data distribution does not match well the true data distribution, or an unnaturally high number of clusters is required to represent the true data distribution of the data set. In this paper we propose OCI (Outlier-robust Clustering using Independent Components), a clustering method which overcomes these problems by (1) applying the exponential power distribution (EPD) as cluster model which is a generalization of Gaussian, uniform, Laplacian and many other distribution functions, (2) applying the Independent Component Analysis (ICA) for both determining the main directions inside a cluster as well as finding split planes in a top-down clustering approach, and (3) defining an efficient and effective filter for outliers, based on EPD and ICA. Our method is parameter-free and as a top-down clustering approach very efficient. An extensive experimental evaluation shows both the accuracy of the obtained clustering result as well as the efficiency of our method.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: Clustering

General Terms

Algorithms

Keywords

Outlier-robust Clustering, ICA, EPD

SIGMOD'08, June 9–12, 2008, Vancouver, BC, Canada.

1. INTRODUCTION

Clustering is an unsupervised learning task that has attracted considerable attention during the past decades. Clustering means to group together objects such that the intra-group similarity is maximized and the between-group similarity is minimized. Multiple books, surveys, and research papers ([10, 18, 3, 6, 17], to name just a few) underline the impact of this research area to the whole research community. In addition, clustering has also been very successfully established in various application areas such as customer segmentation, molecular biology, medical imaging etc.

Recently, the work on clustering has focused on the scalability of the algorithms to large data sets [18], to find clusters of an arbitrary shape (in contrast to a focus on Gaussian clusters in early work) [15], to find clusters in which the grouped objects share complex attribute dependencies [6, 1, 17], and clustering methods that are particularly insensitive to outliers [10]. Moreover, recent work has also been addressing the questions, how to select difficult parameter settings such as the number of clusters to be searched, e.g. [11], or to do this setting fully automatically, or to completely avoid such parameters. And, finally, modern clustering methods give an informative and concise description of the cluster content, e.g. in terms of a multivariate probability distribution function (PDF).

For the description of the cluster content, in most conventional approaches such as K-Means and EM-clustering [8] but also more recent approaches such as G-Means [12] or X-Means [16], a Gaussian model is used which corresponds to the well-known PDF of the normal distribution:

$$f_{Gauss[\vec{\mu},\Sigma]}(\vec{x}) = \frac{1}{\sqrt{2\pi \cdot \det(\Sigma)}} \mathbf{e}^{(-\frac{1}{2}(\vec{x}-\vec{\mu})^{\mathrm{T}} \cdot \Sigma^{-1} \cdot (\vec{x}-\vec{\mu}))}.$$

Unfortunately, this cluster model is not flexible and powerful enough to handle many real life data sets which do not follow a Gaussian distribution. However, many of the building blocks of current clustering methods rely on the (often implicit and tacid) assumption of a Gaussian distribution. To find the main directions into which a cluster is extended is typically done by PCA (principal component analysis), by maximizing the variance of the data set. It has been shown, however, that the directions of main variance are sufficient and actually meaningful only for Gaussian data [13].

To find directions which are not only de-correlated but give also statistically completely independent projection vectors in which one is able to find more sophisticated probability distributions (such as uniform distributions, Laplacian distributions but also multi-modal distributions, i.e. direc-

tions of possible split axes), we need the Independent Component Analysis (ICA). The center point of the PDF describing a cluster content is typically selected as the empirical mean (center of gravity) of the associated points ($\vec{m} = 1/|C| \cdot \sum_{\vec{x} \in C} \vec{x}$). This is also useful for Gaussian data only. It is well-known that for data following a Laplacian distribution the median should be selected, and for the uniform distribution, the mean between the maximum and minimum element (not the mean of all elements) should be used. There are many more implicit and tacit Gaussianity assumptions in current clustering methods. For instance, when a point is associated to that cluster representative which is closest (according to Euclidean or Mahalonobis distance), we implicitly have assumed that the data distribution of the clusters is Gaussian.

Another assumption of many previous clustering algorithms is that every objects indeed belongs to one of the clusters, and that real outlier objects which do not fit well to any of the clusters do not exist. We will show in this paper that every applied estimation method of cluster representatives, main directions, and the associated probability density functions is very sensitive with respect to these outliers, particularly if the outliers are very few (because then they are not combined into a cluster) and if they are far away from the actual clusters (because then they have a large influence on statistical measures of the data like the *mean*). Therefore, it is necessary to carefully handle outliers in a clustering algorithm.

In this paper, we propose **OCI (Outlier-robust Clustering using ICA)**, a novel, efficient, scalable, and fully automatic clustering method beyond the Gaussianity assumption. OCI is based on the Independent Component Analysis (ICA), and uses this method in two of its building blocks, (1) to identify suitable directions for splitting a cluster into two sub-clusters in a top-down clustering approach and (2) to identify the major directions inside a cluster which can successfully be described by more powerful distribution functions than the Gaussian distribution.

For the purpose of describing the cluster content, we use the family of Exponential Power Distributions (EPD). This is a generalization of many distribution functions and contains, as special cases, even the uniform, Gaussian, and Laplacian distribution but also an infinite number of platikurtic (sub-Gaussian) and leptokurtic (super-Gaussian) distributions. The EPD has a shape parameter which controls the Gaussianity (kurtosis) of the distribution. All parameters of an EPD can be efficiently estimated using Maximum Likelihood Estimation (MLE).

In addition, we propose an efficient but powerful method to filter out outliers of the data set which is closely integrated into our process of ICA and MLE. An outlier is hereby defined as an object that does not fit well into any of the distribution functions associated with the clusters. Our method iteratively applies outlier filtering and redetermining of the PDF in order to ensure that the PDF is in the least possible way influenced from the outliers and vice versa. The major contributions of our approach can be summarized as follows:

- Based on a very general cluster notion supported by EPD and ICA, our algorithm is applicable on a wide variety of data distributions including e.g. Gaussian, Laplacian, uniform.
- We propose an efficient and effective de-noising technique, which makes our algorithm even applicable to data containing a high percentage of noise.
- Our algorithm is theoretically funded on the idea to minimize the entropy of the resulting clustering.
- Besides ICA, OCI employs strategies derived from the Minimum Description Length principle to fully automatically find a clustering with low entropy.

The paper is organized as follows: In the next section, we briefly survey related work. In Section 3 we elaborate the OCI algorithm starting with introducing a flexible powerful cluster model supported by ICA and EPD. In Section 4 we derive effective and efficient algorithms for filtering outliers. In Section 5 we elaborate how we can put it all together ending up with an efficient fully automatic top-down clustering algorithm. Section 6 provides an extensive experimental evaluation and Section 7 concludes the paper.

2. RELATED WORK

During the last decades a large variety of clustering algorithms have been proposed, e.g. BIRCH [18], CLIQUE [2], ORCLUS [1], DBSCAN [10] and X-Means [16]. However, they all suffer from one or more of the following drawbacks: The cluster model is restricted to Gaussian, or the result of the algorithm is strongly affected by single outliers, or the algorithm is highly sensitive to parameter settings, and/or the algorithm provides no model summarizing the characteristics of the data. This section provides a brief summary on related approaches intended to overcome at least one of the mentioned drawbacks.

Clustering with Probability Density Functions. The idea of describing the cluster structure of a data set by a mixture model of probability density functions goes back to the widespread EM clustering algorithm [8]. Starting with an arbitrary initialization, the algorithm iteratively optimizes a mixture model of k Gaussian distributions until no further significant improvement of the log-likelihood of the data can be achieved, which is controlled by a threshold for convergence. Usually, a very fast convergence is observed. The resulting Gaussian mixture model provides valuable information for the interpretation and utilization of the result. However, the algorithm can get stuck in a local maximum of the log-likelihood. Moreover, the quality of the clustering result strongly depends on an appropriate choice of the number of clusters k, which is a non-trivial task in most applications. Even with a suitable choice of k the algorithm is very sensitive w.r.t. noise and outliers.

Density-based Clustering. For clustering data sets containing a large percentage of noise points, the DBSCAN algorithm [10] is much more suitable than EM. DBSCAN relies on a density-based clustering notion which is very robust w.r.t. outliers. Areas of high object density which are surrounded by areas of lower object density are regarded as clusters. A density threshold for clustering is defined by two parameters: ϵ specifying a volume and $MinPts$ specifying a minimum number of objects. An object is called *core object* if at least $MinPts$ objects are within a distance of ϵ. Two objects are called *density connected* if they are linked by a chain of core objects. Starting with an arbitrary core object, a cluster can be detected by collecting all density connected objects. As an extension of DBSCAN, the

OPTICS algorithm [3] derives the clusterings w.r.t. various density thresholds without affecting the overall runtime complexity which is quadratic in the number of objects. As a result, the so-called *reachability plot* illustrates the hierarchical cluster structure of the data set. Density-based clustering algorithms detect arbitrarily shaped clusters of arbitrary data distribution. However, no model is provided which is often essential for the utilization of the clustering result. The other problem is again the appropriate parametrization. The result of DBSCAN strongly depends on the density threshold. Also for OPTICS, if a non-hierarchical flat clustering is desired, it is often difficult to decide at which density threshold the clusters should be extracted from the reachability plot.

Parameter-free Clustering. Some approaches preliminarily focus on avoiding crucial parameter settings in clustering. For model-based clustering, such as EM and K-Means there exists a trade-off between the complexity of the model and its quality for data description and interpretation. A maximal complex model perfectly captures the data distribution but provides no novel insight since it is equally complex as the data itself. On the other hand, a model of too low complexity may miss important trends in the data. The user specifies the model complexity by parameter settings, most importantly by selecting the number of clusters k. Most approaches to parameter-free clustering, e.g. X-Means [16], G-Means [12] and RIC [5] employ information-theoretic criteria to achieve a balance between the complexity of the model and its quality for interpretation. However, these approaches rely on a relatively simple cluster notion. As an extension of K-Means, X-Means is restricted to spherical Gaussian clusters. G-Means can additionally detect clusters exhibiting linear attribute correlations, but is still restricted to Gaussian distributed data. Both, X-Means and G-Means are very sensitive w.r.t. outliers due to the underlying K-Means algorithm. RIC has been designed as a postprocessing step to improve an initial clustering of an arbitrary conventional clustering algorithm. After filtering the initial clusters from noise points, for each cluster a model is determined. This model comprises a rotation matrix determined by PCA and a PDF assigned to each coordinate selected from a set of predefined PDFs. Clusters with similar characteristics are finally merged. RIC achieves to improve an imperfect initial clustering by partially correcting the consequences of the limitations of the initial clustering algorithm and bad parameter settings. However, the final result strongly depends on the quality of the initial clustering and the cluster model is limited to linear attribute correlations and a predefined set of PDFs.

Correlation Clustering. Correlation clustering algorithms such as 4C [6] have been designed to detect clusters exhibiting linear attribute dependencies. The 4C algorithm relies on the density-based cluster notion of DBSCAN. Correlation clusters are defined as dense areas of the data space exhibiting linear attribute correlations. CURLER [17] detects non-linear correlation clusters with a two-step procedure: First a large set of so-called microclusters are computed using the EM-algorithm. Microclusters with similar characteristics, e.g. similar orientation are merged in the second step. CURLER and 4C provide no model of the data.

3. CLUSTER MODEL

3.1 What Is a Good Cluster?

The goal of this section is to develop a cluster notion, representation and description method, which is powerful, and can handle both Gaussian and non-Gaussian clusters. This cluster description should provide a good summarization of the cluster content in terms of a probability distribution function. In addition to the goal of providing a good summarization, the cluster representation has to facilitate the process of separating clusters from each other and to identify (and filter out) outlier objects which have wrongly been assigned to a cluster.

The traditional way to represent clusters in clustering methods such as EM clustering or K-Means is the use of Gaussian distribution functions, the density function of which is defined as follows:

$$f_{Gauss[\vec{\mu},\Sigma]}(\vec{x}) = \frac{1}{\sqrt{2\pi \cdot \det(\Sigma)}} \mathbf{e}^{(-\frac{1}{2}(\vec{x}-\vec{\mu})^{\mathrm{T}} \cdot \Sigma^{-1} \cdot (\vec{x}-\vec{\mu}))}.$$

Here, $\vec{x}$ represents a point object from a d-dimensional vector space, $\vec{\mu}$ is the center of the cluster and Σ is a covariance matrix which can also be restricted to diagonal matrices (i.e. axis-parallel Gaussian clustering of some EM clustering variants) or the identity matrix I, the implicit model of K-Means clustering. But the more general model of Gaussian clusters is to allow attribute dependencies, which are modelled in a (non-diagonal) matrix Σ, resulting in Gaussian curves which are not axis-parallel. Given a set of points, Σ corresponds to the covariance matrix of the objects belonging to the cluster, and the main directions can be determined by the Principal Component Analysis (PCA), the Eigenvalue decomposition of Σ.

However, for many clusters in real-life data sets, the Gaussian model is not a sufficient description and representation of the cluster content. Some or all of the coordinates of the cluster may follow a non-Gaussian distribution, for instance the uniform distribution, or the Laplacian distribution. Or a coordinate is a mixture which has been created by a linear combination (e.g. weighted sum) of some distribution functions.

3.2 The Exponential Power Distribution

A large class of distribution functions including prominent functions such as Gaussian, Laplacian, and uniform, but also an infinite number of further distribution functions is the exponential power distribution (EPD). In the uni-variate case, the density function of the EPD is defined using three parameters, the location parameter μ, the scale parameter σ, and the shape parameter p [14]:

$$f_{EPD[\mu,\sigma,p]}(x) = \frac{1}{2\sigma p^{1/p}\Gamma(1+1/p)} \mathbf{e}^{(-\frac{|x-\mu|^p}{p\sigma^p})},$$

where $\Gamma(s)$ is the gamma function $\Gamma(s) = \int_0^\infty t^{s-1} \cdot \mathbf{e}^{-t}\mathbf{d}t$ which is the extension of the factorial operator for real numbers (i.e. $\Gamma(s) = (s-1)!$ for $s \in \mathbb{N}$). The shape parameter p determines the shape of the distribution function and corresponds to a Gaussian distribution for $p = 2$, a Laplacian distribution for $p = 1$ and a uniform distribution for $p \to \infty$. In general, p determines the *kurtosis* of the distribution function: With $p > 2$, the EPD is platykurtic (more flat than a Gaussian distribution) and for $0 < p < 2$ it is leptokurtic. Instead of directly providing the parameter p

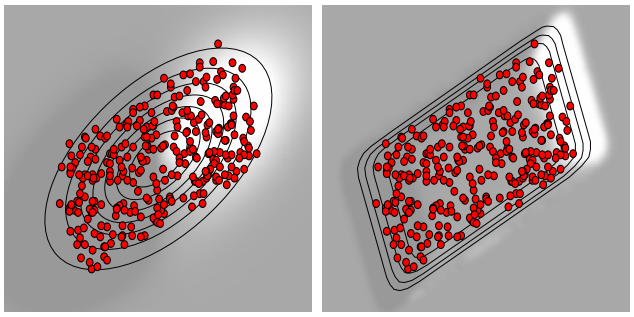

Figure 1: PDF as Gaussian (l.) and EPD (r.).

which is the exponent applied to the data objects, sometimes alternatively a parameter $\beta = 2/p - 1$ is defined. This representation of the shape parameter is somewhat more intuitive from an application viewpoint because the Gaussian distribution forms the zero line ($\beta = 0$) and leptokurtic distributions have $\beta > 0$ (with the Laplacian distribution $\beta = 1$) and platykurtic distributions have $\beta < 0$ (e.g. the uniform distribution with $\beta = -1$). Distributions beyond the Laplacian distribution ($\beta > 1$, the so-called fractional distributions) are also possible and meaningful. We will refer in our experiments to the β-representation of the shape parameter.

For the multivariate case, it is typically assumed that there are d different distribution functions $f_{EPD[\mu_i,\sigma_i,p_i]}(z_i)$, with $1 \leq i \leq d$ all following an exponential power distribution, which are combined by a mixing matrix M, i.e. the observed vectors $\vec{x}$ correspond to $\vec{x} = M \cdot \vec{z} + \vec{m}$ where z_i follows the distribution $f_{EPD[\mu_i,\sigma_i,p_i]}$. The vector $\vec{m}$ is a shift vector, as we will see in the next section. For Gaussian distributions, it can be shown that it is sufficient to assume an orthonormal matrix for M (which can be determined by PCA). For leptokurtic and platykurtic distributions, this simplification is not possible and we have to allow more general mixing matrices in which the vectors indicating the so-called independent components are not orthogonal. Given a d-dimensional point $\vec{x}$, the value of the probability density function at the point can be determined as follows:

$$f_{EPD[\vec{m},M^{-1},\vec{\mu},\vec{\sigma},\vec{p}]}(\vec{x}) = \frac{1}{|\det(M^{-1})|} \cdot \prod_{1 \leq i \leq d} f_{EPD[\mu_i,\sigma_i,p_i]}(z_i),$$

where $\vec{z} = M^{-1} \cdot (\vec{x} - \vec{m})$. We will see later (Section 3.5) that the parameters $\vec{m}$ and $\vec{\mu}$ play a similar role, and can be integrated into one common parameter vector called $\vec{\mu}$ to save redundancy. Likewise, the de-mixing matrix M^{-1} and the scale vector $\vec{\sigma}$ can be integrated into one overall de-mixing and shape matrix $\underline{M}^{-1}$. The actual position and shape parameters of the independent univariate EPD distributions are then set to the standard values $\mu_i = 0$ and $\sigma_i = 1$:

$$f_{EPD[\vec{\mu},\underline{M}^{-1},\vec{p}]}(\vec{x}) = \frac{1}{|\det(\underline{M}^{-1})|} \cdot \prod_{1 \leq i \leq d} f_{EPD[0,1,p_i]}(z_i),$$

where $\vec{z} = \underline{M}^{-1} \cdot (\vec{x} - \vec{\mu})$.

The exponential power distribution in combination with the mixing matrix M is a much more powerful description of the content of a cluster compared to the Gaussian cluster model. Consider for instance the data set in Figure 1 in which both coordinates are combinations of almost uniform distributions. Provided that a good estimate of the main directions and the parameters of the distribution can be determined (cf. Section 3.4), EPD gives an exact and concise representation of the cluster which can be used e.g. for estimating the number of points which are enclosed in a given volume. In contrast, the Gaussian model is less accurate, and we will formalize in the next section the corresponding notion of accuracy.

3.3 Quality of a PDF and Data Compression

In the last section, we have visually compared two distribution functions and have recognized that for a given sample data set the representation by an EPD function may be a more exact representation of the data than the Gaussian PDF. Now, we have to determine a more formal way to assess the accuracy of a distribution function. A well established tool for this assessment is the principle of the Minimum Description Length (MDL) which links the concept of PDFs to data compression. Data compression using Huffman coding assigns a number of bits to each object which is the logarithm of the inverse of the probability (the negative *log-likelihood*) of the object, i.e. the coding cost of a data object, Huffman-coded with a given PDF is:

$$c_{PDF}(\vec{x}) = \log_2(\frac{1}{f_{PDF}(\vec{x})}) = -\log_2(f_{PDF}(\vec{x})).$$

The basis of the logarithm is typically 2 to represent the coding cost in *number of bits*. To get an absolute value of the number of bits, it is necessary to select a grid resolution which defines the accuracy with which the points have to be stored. Since one is only interested in comparing different PDFs the selection of such a grid is not necessary but we want to note that these relative costs may also have negative values.

If there are two candidate PDFs (PDF_1 and PDF_2) to represent the data, we can compare the cost for the whole cluster and conclude that for instance PDF_1 is a more exact representation of the data than PDF_2 if

$$\sum_{\vec{x} \in C} c_{PDF_1}(\vec{x}) < \sum_{\vec{x} \in C} c_{PDF_2}(\vec{x}).$$

In our case of an exponential power distribution defined by a mixing matrix M and an individual EPD with parameters (μ_i, σ_i, p_i) for each dimension $i \in \{1, ..., d\}$ we obtain:

$$c_{EPD}(\vec{x}) = \log_2(|\det(M^{-1})|) - \sum_{1 \leq i \leq d} \log_2(f_{EPD[\mu_i,\sigma_i,p_i]}(z_i)),$$

where $\vec{z} = M^{-1} \cdot (\vec{x} - \vec{m})$.

To really be able to decompress the data again, one also needs to have the *codebook* of the data, i.e. the parameters of the PDF. The number of bits required to represent the codebook must also be added to the overall cost. Moreover, for each object we have to assign the information to which cluster C it belongs (in order to decide which codebook has been applied). This information can also be Huffman-coded using $\log_2(n/|C|)$ bits.

3.4 ICA and MLE of EPD

Given the set of data vectors $\vec{x} \in C$ belonging to a cluster C, there remain two problems to find a good cluster rep-

resentative, i.e. (1) to find the mixing matrix M (or the *de-mixing* matrix M^{-1}, respectively) which transforms the coordinates of $\vec{x}$ into independent coordinates $\vec{z}$ which can be represented by an EPD function in an optimal way, and (2) to determine the parameters $[\mu_i, \sigma_i, p_i]$ with $1 \leq i \leq d$ of the EPD. From the theory of independent component analysis, it follows that the mixing matrix can be found by determining the directions of minimal entropy (in contrast to the directions of maximal variance, which are searched by PCA). To find the directions of minimal entropy, the well-known fastICA algorithm [13] requires to transform the data objects into the so-called white space, i.e. the data must be centered ($= \vec{x} - \vec{m}$ where $\vec{m} = \frac{1}{|C|}\sum_{\vec{x}\in C}\vec{x}$ is the *empirical* mean of the data set C) and normalized to have unit variance in all directions. This may be done from the eigenvalue decomposition of the covariance matrix (i.e. $V \cdot \Lambda \cdot V^{\mathbf{T}} := \Sigma$ where V is an orthonormal matrix consisting of the eigenvectors and Λ is a diagonal matrix consisting of the eigenvalues of Σ) by setting $\vec{y} := \Lambda^{-1/2} \cdot V^{\mathbf{T}} \cdot (\vec{x} - \vec{m})$. Since Λ is a diagonal matrix ($\Lambda = \mathrm{diag}(\lambda_1, ..., \lambda_d)$), the matrix $\Lambda^{-1/2}$ is the diagonal matrix with the elements $\Lambda^{-1/2} = \mathrm{diag}(\sqrt{1/\lambda_1}, ..., \sqrt{1/\lambda_d})$. The fastICA algorithm then determines a matrix W which contains the independent components. This matrix is orthonormal in white space but not in the original space. FastICA is an iterative method that finds $W = (\vec{w}_1, ..., \vec{w}_d)$ by optimizing the vectors $\vec{w}_i$ by the following updating rule:

$$\vec{w}_i := \mathrm{E}\{\vec{y} \cdot g(\vec{w}_i^{\mathbf{T}} \cdot \vec{y})\} - \mathrm{E}\{g'(\vec{w}_i^{\mathbf{T}} \cdot \vec{y})\} \cdot \vec{w}_i,$$

where $g(s)$ is a non-linear contrast function (such as $\tanh(s)$) and $g'(s) = \frac{d}{ds}g(s)$ is its derivative. By $\mathrm{E}\{...\}$, we denote the expected value. After each application of the update rule to $\vec{w}_1, ..., \vec{w}_d$, the matrix W is orthonormalized. This is repeated until convergence. The de-mixing matrix M^{-1} which describes the overall transformation from the original data space to the independent components can be determined as

$$M^{-1} = W^{\mathbf{T}}\Lambda^{-1/2} \cdot V^{\mathbf{T}}, \text{ and } M = V \cdot \Lambda^{+1/2} \cdot W$$

and, since V and W are orthonormal matrices, the determinant of M^{-1} is simply the determinant of $\Lambda^{-1/2}$, i.e.

$$\det(M^{-1}) = \prod_{1\leq i\leq d}\sqrt{1/\lambda_i}.$$

The result of the fastICA algorithm is unambiguous if not more than one component follows a Gaussian distribution. If the data is normally distributed in two or more directions, then the algorithm picks the corresponding components more or less randomly among the Gaussian directions. We would like to note, however, that it can be shown that the overall multivariate distribution function is independent from the selection of these components, i.e. if different components can be selected, they are different but equivalent representations of the same PDF.

Let us further note that the FastICA algorithm implicitly searches for those directions in the data space, in which the data distribution is as dissimilar to the Gaussian distribution as possible. The original motivation for this strategy is that observed signals which are mixtures of original signals can be best de-mixed when searching for non-Gaussianity, because when mixing two or more signals which follow an arbitrary distribution each, then the mixed signal is always more Gaussian than the original ones (where the central limit theorem says that an infinite number of mixed signals is always exactly Gaussian distributed).

But for our purpose of accurately representing the cluster content by a multivariate PDF it is also beneficial to search for directions which are non-Gaussian, because in white space the *entropy* of a Gaussian distribution is maximal (i.e. all other distributions have lower entropy). The entropy is exactly a measure of the coding cost, and, therefore, when we identify the directions which are maximally non-Gaussian then the coding cost is minimized, and the data set can be compressed with maximal compression efficiency.

After transforming the points by $\vec{z} = M^{-1}\cdot(\vec{x}-\vec{m})$, the coordinates z_i are not only de-correlated but also independent in a more general way which allows the description of each coordinate independently by an EPD. But it is also a non-trivial problem to identify the parameters $[\mu_i, \sigma_i, p_i]$ of the EPD distribution function of the de-mixed coordinates. If a coordinate follows a Gaussian distribution, then $p_i = 2$ and μ_i and σ_i can be chosen as the mean and standard deviation of the data, respectively. Since the data are in white space after the application of fastICA (and particularly its preprocessing steps), we know anyway that $\mu_i = 0$ and $\sigma_i = 1$. However, for non-Gaussian data, the location parameter μ_i is not in general identical to the empirical mean of the data set ($m_i = \frac{1}{|C|}\sum_{\vec{z}\in C} z_i$), and the scale parameter σ_i is not identical with the empirical standard deviation of the data set. Instead a maximum likelihood estimation algorithm must be applied to determine an optimal parameter setting given the data set C. The three parameters μ_i, σ_i and p_i must be simultaneously optimized to ensure that the derivatives of the likelihood $\sum_{\vec{z}\in C} f_{EPD[\mu_i,\sigma_i,p_i]}(z_i)$ with respect to μ_i, σ_i, and p_i vanish. It can be shown that σ_i can be determined directly given μ_i and p_i by using

$$\frac{df_{EPD[\mu_i,\sigma_i,p_i]}(C)}{d\sigma_i} = -\frac{|C|}{\sigma_i} + \frac{1}{\sigma_i^{p_i+1}}\sum_{\vec{z}\in C}|z_i - \mu_i|^{p_i} = 0$$

$$\Rightarrow \sigma_i = (\frac{1}{|C|}\sum_{\vec{z}\in C}|z_i - \mu_i|^{p_i})^{1/p_i},$$

but μ_i and σ_i must be optimized explicitly, e.g. using Newton optimization. We apply a faster and more stable method which applies a nested bisection search to p_i and μ_i, in each step determining the derivatives of the likelihood function. The derivatives of the likelihood function with respect to p_i and μ_i (but not their root) can be determined analytically:

$$\begin{aligned}
\frac{df_{EPD[\mu_i,\sigma_i,p_i]}(C)}{d\mu_i} &= -\frac{1}{\sigma^p}\sum_{\vec{z}\in C}|z_i - \mu_i|^{p_i-1}\,\mathrm{sign}(z_i - \mu_i).\\
\frac{df_{EPD[\mu_i,\sigma_i,p_i]}(C)}{dp_i} &= -\frac{|C|}{p_i^2}\left(\log(p_i) + \Psi(1+\frac{1}{p_i}) - 1\right)\\
&+ \frac{1}{p_i^2\sigma^{p_i}}\sum_{\vec{z}\in C}|z_i - \mu_i|^{p_i}\\
&+ \frac{1}{p_i\sigma^{p_i}}\left(\log(\sigma_i)\sum_{\vec{z}\in C}|z_i - \mu_i|^{p_i}\right.\\
&- \left.\sum_{\vec{z}\in C}|z_i - \mu_i|^{p_i}\log|z_i - \mu_i|\right).
\end{aligned}$$

Here, $\Psi(s) = \frac{\mathbf{d}}{\mathbf{d}s}\ln\Gamma(s)$ denotes the *digamma function*, the logarithmic derivative of the gamma function.

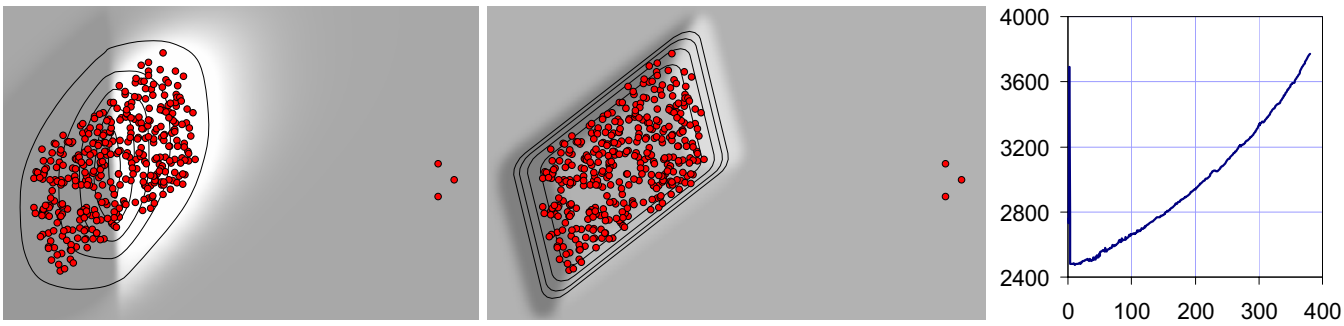

Figure 2: Non-robust (l.) and Robust (m.) Estimation of the PDF and MDL-Cost by Outlier Filtering (r.)

3.5 The Overall ICA-EPD Pipeline

Now, we summarize our overall pipeline to obtain a multivariate PDF which is a mixture of d EPD functions for a given data set C without addressing the problem of outlier filtering (which will be considered in Section 4). Our pipeline starts with *data centering*, i.e. determining the empirical mean $\vec{m}$ of the data set and subtracting $\vec{m}$ from every data object. Then, we *whiten* the data by applying PCA which determines the eigenvector V and eigenvalue matrix Λ of the covariance matrix Σ. The data objects are projected orthogonally to the eigenvectors and then divided by the square root of the eigenvalues which de-correlates the data objects and creates unit variance ($\sigma^2 = 1$) not only of each coordinate but even for every possible linear projection. Then we apply the fastICA algorithm to determine those directions (matrix W) in the white space which can be optimally compressed under the assumption of Huffman coding. The data points are projected onto those independent components. Finally, we perform a *maximum likelihood estimation of the EPD* in each of the obtained coordinates, each of which gives us the three parameters (μ_i, σ_i, p_i) of the EPD.

The location parameter μ_i and the scale parameter σ_i of the EPD play a similar role in the overall pipeline like the mean $\vec{m}$ for centering and the covariance matrix Σ for whitening. Therefore, it is convenient to combine these informations to get one *overall location vector* which represents the actual center of the multivariate PDF and one *de-mixing and shape matrix* which performs all necessary rotations and scalings to transform the data objects into a space where all scale parameters of EPD are unity. In this space, we know that the EPD parameters $\mu_i = 0$ and $\sigma_i = 1$, (i.e. a kind of *standard EPD* is applied), and only the d shape parameters p_i need to be additionally stored in the codebook of the cluster.

The overall location vector $\underline{\vec{\mu}}$ can be determined as follows: In our pipeline, the points have first been centered by moving them by the shift vector $\vec{m}$. Then, after whitening and projecting onto the independent components, a new mean $\vec{\mu} = (\mu_1, ..., \mu_d)^{\mathbf{T}}$ has been determined in the whitened projected space. To find the mean in the original space, we have to undo the projection (by multiplying with the inverse of $W^{\mathbf{T}}$) and undo whitening of $\vec{\mu}$ and obtain

$$\underline{\vec{\mu}} = \vec{m} + V \cdot \Lambda^{+1/2} \cdot W \cdot (\mu_1, ...\mu_d)^{\mathbf{T}}.$$

Analogously, we obtain the overall de-mixing and shape matrix $\underline{M}^{-1}$ by multiplying the diagonal matrix containing the inverted scale parameters $1/\sigma_i$ to the de-mixing matrix M^{-1}:

$$\underline{M}^{-1} = \text{diag}(1/\sigma_1, ..., 1/\sigma_d) \cdot W^{\mathbf{T}} \cdot \Lambda^{-1/2} \cdot V^{\mathbf{T}}.$$

4. OUTLIER-ROBUST ICA AND EPD

In this section, we show how to make both the Independent Component Analysis to estimate the de-mixing matrix M^{-1} as well as the maximum likelihood estimation of the EPD function outlier robust. Outlier robustness is a particular issue here, because all the methods which are usually applied to estimate the center and the shape of a distribution function are very sensitive to outliers. Consider, for instance, the data set which is schematically given in Figure 2 (the outliers are actually meant to be even more distant from the actual cluster than depicted). In this case, the empirical mean of the data set is much (linearly) influenced by the high distance of the outlier from the core cluster. The problem becomes even worse when determining the covariance matrix, because the influence of an outlier to the variance is even quadratic, and we obtain an additional error because the wrongly estimated mean influences the estimation of the covariance matrix as well. But this error is propagated to ICA (requiring the mean and covariance matrix for whitening and being sensitive to outliers due to the nonlinear function) and to the MLE of the EPD (which would try to cover the outlier by its PDF).

To find a good PDF for the data, we first have to filter out the outliers and then can apply ICA and EPD estimation. Some methods for identifying outliers have been proposed, but in our case, the appropriate definition of an outlier is that it does not fit well to the PDF defined by the de-mixing matrix and EPD of the cluster. Therefore, we obtain a circular dependency between the outlier detection at the one hand and the PDF estimation at the other hand: A good outlier filter requires a well estimated PDF and vice versa. We can escape from this circular dependency by applying an iterative method which applies outlier filtering and PDF estimation in an alternating way until convergence. We propose two different methods: the first performing a loop which tentatively removes in the first iteration 1 point from the cluster, then two, and so on. The removed i objects in iteration number i are always those objects which fitted worst to the PDF that was determined in iteration number $(i-1)$. Finally, the partition with best overall coding cost of the data is selected. The second, alternative method for outlier filtering is more related to the K-Means method, and,

```
algorithm LoopwiseOutlierFiltering(set of point C)
  μ⃗ := (μ_1, ..., μ_d)
  M^{-1} := diag(σ_1, ..., σ_d)        } to minimally enclose all obj. of C;
  β⃗ := (−1, ..., −1) ; (* β = −1 ⇔ uniform distribution *)
  currentPdf:=(μ⃗, M^{-1}, β⃗);
  outlierCost:=cost of an object assuming currentPdf;
  bestCost:=+∞;
  (μ⃗, M^{-1}, β⃗) := mleEpd(fastICA(whiten(center(C))));
  currentPdf:=(μ⃗, M^{-1}, β⃗);
  for i:=0 to |C|
    determine cost for every object in C according to currentPdf;
    outliers:=those i objects of C having maximum cost;
    core:=C \ outliers;
    cost:=Σ_{x⃗∈core} cost(x⃗) + i · outlierCost;
    if cost < bestCost then
      bestCost:=cost;
      bestCore:=core;
      bestOutliers:=outliers;
      bestPdf:=currentPdf;
    (μ⃗, M^{-1}, β⃗) := mleEpd(fastICA(whiten(center(core))));
    currentPdf:=(μ⃗, M^{-1}, β⃗);
```

Figure 3: Loopwise Algorithm for Outlier Filtering.

```
algorithm IterativeRepartitioning(set of point C)
  μ⃗ := (μ_1, ..., μ_d)
  M^{-1} := diag(σ_1, ..., σ_d)        } to minimally enclose all obj. of C;
  β⃗ := (−1, ..., −1) ; (* β = −1 ⇔ uniform distribution *)
  currentPdf:=(μ⃗, M^{-1}, β⃗);
  outlierCost:=cost of an object assuming currentPdf;
  (μ⃗, M^{-1}, β⃗) := mleEpd(fastICA(whiten(center(C))));
  currentPdf:=(μ⃗, M^{-1}, β⃗);
  while not converged
    outlier:={};
    core:={};
    for each x⃗ ∈ C do
      if cost(currentPdf, x⃗) < outlierCost then
        core:=core ∪{x⃗};
      else
        outlier:=outlier ∪{x⃗};
    cost:=Σ_{x⃗∈core} cost(x⃗) + i · outlierCost;
    (μ⃗, M^{-1}, β⃗) := mleEpd(fastICA(whiten(center(core))));
    currentPdf:=(μ⃗, M^{-1}, β⃗);
```

Figure 4: Iterative Repartitioning for Outlier Filtering.

thus, much faster than loop-wise outlier filtering: It repartitions in each iteration the data set into an arbitrary number of core and outlier objects, and then updates the PDF which is associated to the core cluster. This is repeated until convergence.

4.1 Loopwise Outlier Filtering

This algorithm, defined as pseudocode in Figure 3, starts by estimating a PDF (i.e. performing the workflow: Centering, whitening, ICA, MLE estimation of the EPD) for the whole data set C obtained by an initial clustering without assuming that any objects are outliers. Then, the log-likelihood is determined for each object. The overall log-likelihood is recorded, and the object having least log-likelihood (i.e. highest coding cost) is tentatively removed from the core set of the cluster and moved to the outlier set of the cluster. Then, this is repeated in a loop of $|C|$ iterations: In iteration number i we take the core set which has been identified in iteration number $(i-1)$ and determine the corresponding PDF by our basic workflow (centering, whitening, fastICA, MLE of the d EPD functions). Again, the log-likelihood is determined for each object (including those objects which have previously been filtered out as outliers), and now those i objects with least log-likelihood are tentatively filtered out as outliers for the next iteration. The overall cost for the partitioning of iteration i is determined by applying the determined PDF to the $(|C|-i)$ tentative core objects and a uniform distribution for the i tentative outlier objects. The uniform distribution is selected such that it minimally covers the complete set C of core and outlier objects. In Figure 2 we can see the overall cost determined in each iteration of the loop over varying i. We can see that the overall cost minimum appears at $i = 3$. The PDF of iteration $i = 3$ and the corresponding partition in core and outlier objects is selected as the final outlier filtering. The runtime complexity of this algorithm is $O(n^2)$, since $n-1$ partitions into core and noise points need to be examined.

4.2 Iteratively Repartitioning Outlier Filter

Like the algorithm of Section 4.1 this algorithm starts with the assumption that all objects are core objects of the cluster and determines the optimal PDF for this setting. The iterative repartitioning algorithm (depicted in Figure 4) then partitions the data set into a tentative core and outlier set without predefining any fixed number of this assignment. Rather we assume that the outliers follow a uniform distribution which is chosen such that it contains the whole set C in a minimal way. The objects are assigned to the core or outlier set depending on the log-likelihood (and, thus, depending on the assignment cost). This assignment is used to determine an optimal PDF in the next iteration. The algorithm repeats the two steps, PDF determination and outlier assignment until convergence. The final PDF and partition is the output of the algorithm.

It can be easily shown that the iterative repartitioning algorithm always converges: The overall coding cost of the data set is improved in each iteration of the algorithm, because both steps, reassignment as well as the PDF determination can only improve the cost from one iteration to the next. The reassignment can never increase the cost, because an object changes its assignment if (and only if) this change improves its cost (and, thus, the overall cost). The PDF redetermination can also never increase the cost because both ICA and MLE of EPD *maximize* the overall likelihood of the core set (while the likelihood of each outlier object remains constant). The optimization space (i.e. the parameters) of the current PDF also contains the PDF of the last iteration. Therefore, if no better PDF (improving the cost) exists, then the PDF of the last iteration must be optimal in the current iteration, and in both cases, no cost increase occurs. Since in each iteration the cost is never increased, and the cost are bounded below (by the true data distribution), the convergence is guaranteed. Note that the fastICA algorithm may sometimes fail to identify the true cost minimal components due to numerical problems. In this case, the guarantee of nonincreasing cost cannot be given,

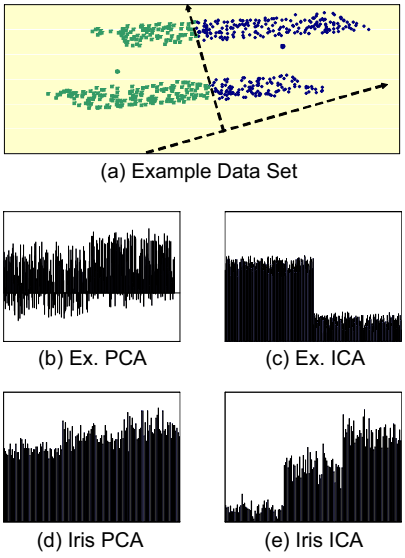

Figure 5: Comparison of Splitting Directions: ICA picks directions with lowest entropy providing 'sparse coding'.

and the convergence is not sure. This problem, however, can easily be cured by explicitly considering the independent components of the fastICA algorithm of the previous iteration again, and only keeping the new result if, indeed, a cost improvement is achieved. The runtime complexity is $O(n \cdot \text{iter})$, whereas iter denotes the number of iterations. We will demonstrate in the experimental section that iteratively repartitioning provides a major speedup without loosing accuracy.

```
algorithm OCI (data set DS)
//Initialization: all objects in one cluster.
  C := {C_start};
  for each object x ∈ DS
    x.clusterID = start;

//Splitting. For details cf. Section 5.1.
  while improvement
    for each C ∈ C
      split(C);

//Outlier Filtering. For details cf. Section 4.
  for each C ∈ C
    clusters core, outliers:= iterativePartitioning(C);
    C = (C \ {C}) ∪ core ∪ outliers;

//Merging. For details cf. Section 5.2.
  while |C| > 1 and improvement
    mergeBestPair(C);
end algorithm

subroutine splitCluster(cluster C):improvement
// Choose splitting direction with best cost: 2-Means or IcaSplit.
  clusters KM_1 and KM_2: = 2-Means(C);
  clusters ICA_1 and ICA_2: = IcaSplit(C);
  clusters minClusters: = minCost((KM_1 ∪ KM_2), (ICA_1 ∪ ICA_2));
  if cost(minClusters)< cost(C)
    C = (C \ {C}) ∪ minClusters;
    return true;
  else return false;

subroutine mergeBestPair(clusters C): improvement
// Find cluster pair with the best (maximal) savedCost).
    for all cluster pairs (C_i, C_j) ∈ C × C
      mergedCost(C_i,C_j) := cost(C_i ∪ C_j);
      savedCost(C_i,C_j):=(cost(C_i)+cost(C_j))-mergedCost(C_i,C_j);
    if argmax_(C_i,C_j)(savedCost(C_i, C_j)) > 0
    merged:= C_i ∪ C_j;
      C = (C \ {C_i, C_j}) ∪ merged;
      return true;
    else return false;
```

Figure 6: OCI Algorithm.

5. TOP-DOWN CLUSTERING ALGORITHM

In this section, we elaborate the OCI algorithm based our general cluster notion supported by outlier-robust ICA and EPD. We start with an effective splitting algorithm to derive initial clusters, which are then purified from outliers using the algorithms introduced in the previous section. To achieve the final result, we merge clusters with similar characteristics, which is described in detail in Section 5.2.

5.1 Splitting

In this section, we discuss the splitting algorithm. In particular, we explain how we can use ICA to find appropriate directions for splitting. Our algorithm may look at first glance similar to conventional top-down splitting algorithms, such as described in X-Means [16] and G-Means [12]. However, in contrast to these algorithms, our method is suitable for non-Gaussian data.

Bisecting K-Means is an attractive building block for a top-down splitting algorithm because it is very efficient. The X-Means algorithm based on bisecting K-Means shows good performance on data sets with spherical Gaussian clusters. G-Means extends X-Means by splitting in each iteration the most non-Gaussian cluster (which is determined by the Anderson-Darling test for Gaussianity), such that also non-axis parallel Gaussian clusters can be detected. The split is performed by bisecting K-Means as in X-Means.

For our more general cluster notion, bisecting K-Means is not so suitable because it converges towards the direction of the largest variance. In [9] it has been proven that the solution of bisecting K-Means is equivalent to the eigenvector with the largest eigenvalue in PCA. For non-Gaussian data, such as the example in Figure 5(a), the direction of largest variance may not be a favorable splitting direction. Figure 5(a) depicts a synthetic data set with 600 points composed of two clusters of 300 points each and a rather uniform data distribution. A typical result of bisecting K-Means is visualized by the cluster centers and coding the cluster assignments in different colors. The eigenvector with the largest eigenvalue is drawn with a dashed line and also the orthogonal direction to this vector is displayed, which corresponds to the split direction of bisecting K-Means.

In Figure 5(b), the projection of the data on the split direction chosen by bisecting K-Means and PCA is displayed. For each object $\vec{x}$ the projection $\Pi_{pca}(\vec{x})$ is computed as follows: $\Pi_{pca}(\vec{x}) = \vec{x} \cdot \vec{v_{max}}$, where $\vec{v}_{max}$ is the eigenvector with the strongest eigenvalue. In Figure 5(b) for each of the 600 objects on the x-axis a bin representing the value of $\Pi_{pca}(\vec{x})$ is displayed on the y-axis, where the objects are ordered w.r.t. their true cluster assignment, i.e. first the 300 objects of cluster on top in Figure 5(a) are displayed followed by the objects of the bottom cluster. It is evident

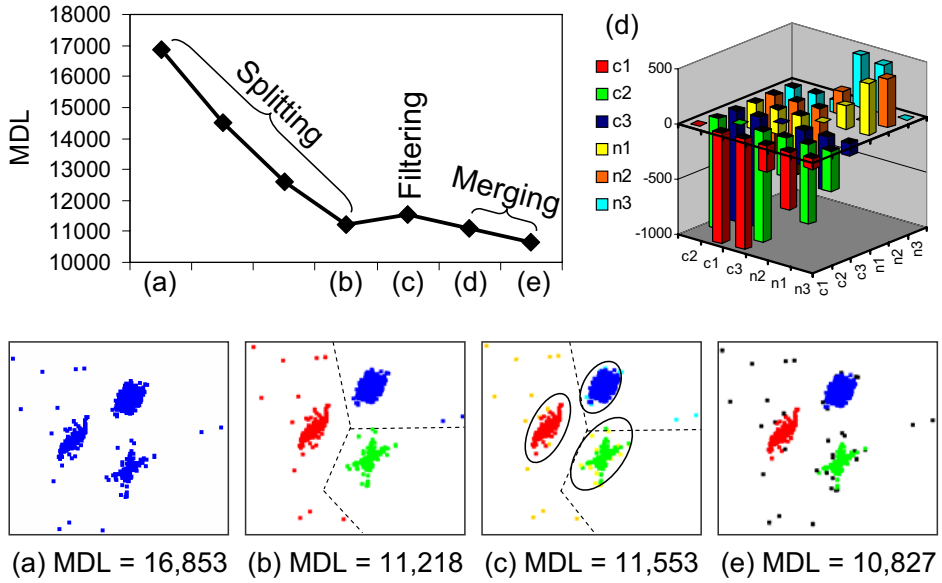

Figure 7: Overview on the Workflow of OCI. The steps splitting, outlier filtering and merging are illustrated on a synthetic example data set. The Minimum Description Length (MDL) curve is visualized together with some important intermediate results. For details see Section 5.2.

that the cluster assignment can not be interfered from Figure 5(b). We propose to employ ICA to guide the top-down algorithm towards better splits. However, since we have no measure for the strength of the independent components as the eigenvalues in PCA, it is a non-trivial question to decide which component should be used as splitting direction. The combination of ICA with EPD provides an interesting option. ICA is guided towards non-Gaussian directions, which can be super-Gaussian as well as sub-Gaussian. Our EPD model allows to exactly specify the degree of super- and sub-Gaussianity by the shape parameter β. For splitting we are only interested to find sub-Gaussian directions, since good splitting directions are such directions in which the data is as much separated as possible. Such directions can be best modelled by uniform distributions. In Figure 5(c) the projection $\Pi_{ica}(\vec{x})$ of the data on the most sub-Gaussian component is displayed. This projection is defined by $\Pi_{ica}(\vec{x}) = \vec{x} \cdot M \cdot \vec{w}_{max}$, where w_{max} denotes the most sub-Gaussian direction, i.e. the direction with the minimal shape parameter β of the EPD. Obviously, both clusters are well separated w.r.t. this direction. Also, on real-world data sets it is often favorable to split along the most sub-Gaussian direction. On the iris data set from the UCI machine learning repository [4] the projection on the first principal component displayed in Figure 5(d) is by far less distinctive than the most sub-Gaussian component, cf. Figure 5(e). In fact, the projection Π_{ica} perfectly separates the three classes of the iris data set. The diagram in Figure 5(e) displays the instances of the classes "iris setosa", "iris viginica" and "iris versicolor" in sequential order. The classes are well separated in the ICA projection, and it is also obvious that iris virginica and iris versicolor are the most similar classes.

ICA split is not in all cases the better choice, e.g. ICA fails if the independent components exhibit a Gaussian distribution, which is likely for higher dimensional large data sets during the first few stages of top-down splitting. Therefore our algorithm selects upon each split the splitting direction maximizing the log-likelihood of the data, which can be either the splitting direction determined by ICA or the direction determined by bisecting K-Means.

5.2 Merging

After splitting, we apply the noise filtering methods proposed in Section 4 to separate an initial cluster in two subsets *core* and *outliers* which are afterwards considered as two different clusters. In particular the outlier sets of some of the original clusters may actually have been generated by the same data distribution, and, therefore, the clustering may become less redundant if such clusters are merged again. Our cluster merging procedure is an iterative process. At each iteration, the algorithm merges those two clusters for which the overall cost gain is maximized. To do so, the cost gain for every pair of clusters is computed by tentatively merging the clusters and reperforming the ICA-EPD pipeline. This step is repeated until no more cost improvement is achieved. Figure 6 summarizes the complete OCI algorithm in pseudocode.

Figure 7 provides an overview of the OCI framework involving the steps splitting, outlier-filtering and merging. The MDL-curve is displayed and each important step is annotated and illustrated on a synthetic data set. Point (a) represents the situation when the MDL-criterion is evaluated the first time at the beginning of the algorithm. All points are assumed to be in a single cluster leading to a MDL value of 16,853. During splitting (three splits in total are performed on this example) the MDL value is drastically reduced. Point (b) depicts the result after splitting with a MDL value of 11,218. After outlier filtering, the MDL value slightly increases to 11,553 at point (c). This is due to the fact that the costs for parameter coding have doubled by introducing three noise clusters. Now, the cost matrix for

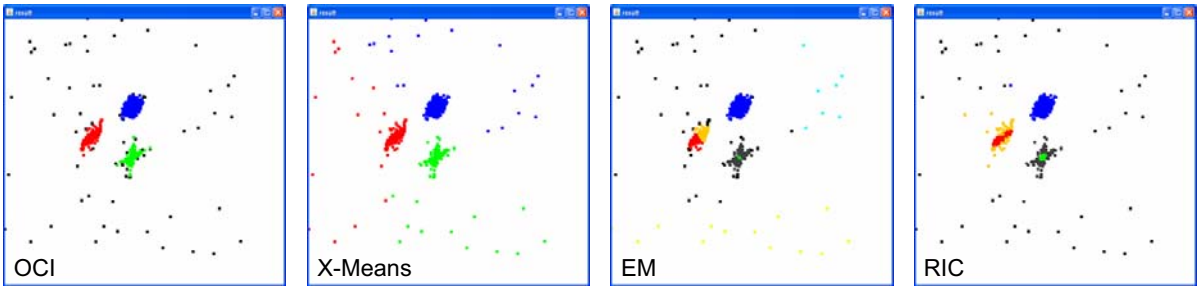

Figure 8: Comparison of OCI to X-Means, EM and RIC on Synthetic Data.

cluster merging is constructed which is visualized in Figure 7(d). It is evident that merging of the noise clusters (denoted by n1-n3) results in savings in coding cost, whereas merging of the core clusters (c1-c3) would lead to an increase in description length. Thus, the noise clusters are greedily merged in two steps. The final result with MDL of 10,827 is depicted at point (e).

6. EXPERIMENTS

In this section we provide a comparison among OCI, EM, RIC and X-Means on synthetic data and on different real-world data sets available at the UCI machine learning repository [4]. For X-Means and EM we used the implementations of the WEKA machine learning package available at `http://www.cs.waikato.ac.nz/ml/weka`. For EM the best number of clusters is obtained by cross-validation, which is implemented in WEKA as follows. The number of clusters is initially set to one. Then, the following steps are iterated: (1) The data is randomly split into 10 folds. (2) EM is performed 10 times on the 10 folds and the log-likelihood is averaged over all results. (3) If the log-likelihood is increased, the number of clusters is increased by one and the program continues with step (1). We we used RIC on top of K-Means with $k = 8$, as suggested by the authors. For X-Means and OCI no parameter settings are required. OCI and RIC have been implemented in Java and experiments were performed on a Linux workstation with a 2.4 GHz CPU and 4 GB main memory.

6.1 Effectiveness on Synthetic Data

We start with a synthetic example first introduced in Figure 7 consisting of 3 non-Gaussian clusters (500 points each) with non-orthogonal major directions and 50 outliers. The results of OCI, EM, X-Means and RIC are depicted in Figure 8. OCI correctly identifies the three clusters and succeeds in filtering out the outliers. The performance of the other algorithms is heavily affected due to the 3.2% noise, which is a relatively small amount, and due to the non-Gaussian major directions of the clusters. From Figure 8 it is evident that X-Means is not suitable for noisy data sets, since all noise points are assigned to the clusters. The BIC-criterion used in X-Means does not favor further splitting to separate the clusters from noise. The cross-validation of EM suggests 8 clusters in total. With the cross-validation option, EM can partially deal with outliers, since additional clusters can be created if this increases the log-likelihood of the data. However, many superfluous clusters are generated by this strategy, which have no semantic meaning in the context of real applications. In addition, it becomes obvious on this example that the assumption of Gaussianity leads to incorrect cluster assignments. The result of RIC shows similar characteristics. Specially designed for parameter-free clustering of noisy data, RIC can cope better with outliers than EM, without creating many unnecessary clusters. However, since the noise filtering algorithm of RIC assumes clusters with orthogonal major directions which can be detected by PCA, RIC fails to correctly filter the clusters in this example.

Figure 9 provides a detailed view on cluster 1 generated with $\beta = -0.85$ for the x-coordinate and $\beta = 3.8$ for the y-coordinate. OCI is the only method detecting this cluster correctly with 99.60% precision and 99.20% recall. X-Means fails to filter the noise, such that the whole Voronoi cell containing cluster 1 is clustered together, leading to a recall of 100%, but only 96.7% of precision. EM represents the core of the cluster as a dense spherical Gaussian, resulting in a low recall of only 51%. RIC favors the model of a linear correlation cluster determined by PCA, which also leads to a reduced recall of 90%. The cluster models of the comparison methods are not rich enough to correctly capture the data distribution of non-Gaussian clusters with non-orthogonal major directions. For this, outlier-robust ICA and a more general class of distribution functions, like EPD is needed.

Also for cluster 2 depicted in Figure 10, OCI is the only algorithm which correctly detects the major directions and the characteristics of the data distribution which is heavy-tailed in both coordinates with $\beta = 4.5$ and 5.5, resulting in a precision of 99.8% and a recall of 97.20% which is the best result among the comparison methods. X-Means again fails to filter the outliers (not depicted), leading to a reduced precision of 97.45%, RIC and EM lack in recall. RIC achieves a recall of 77.2% on cluster 2, EM only retrieves 50.8% of the cluster points. Cluster 3 (not depicted) is more Gaussian in shape, so that the limitations of RIC and EM do not become so apparent.

6.2 Outlier-robustness

To compare the two algorithms for outlier filtering proposed in Section 4 we extracted the 500 points of cluster 2 (cf. Figure 10) and added various amounts of outliers. Figure 11(a) displays a comparison of the loopwise and iterative repartitioning outlier filtering in terms of precision w.r.t. the amount of added outliers ranging from 50 to 600. Figure 11(b) displays the effect on recall. Both algorithms demonstrate similar performance with excellent balance of precision and recall (more than 95% in precision and re-

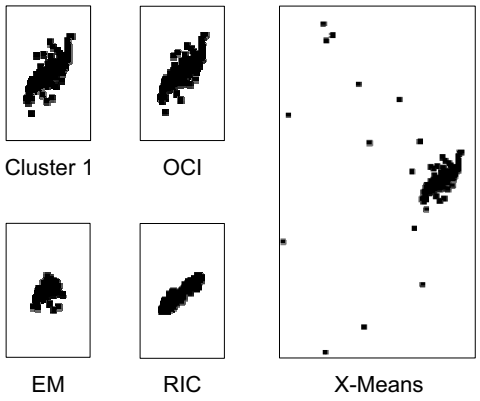

Figure 9: Detailed View of Cluster 1.

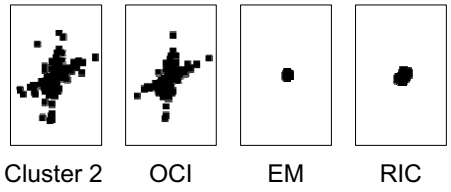

Figure 10: Detailed View of Cluster 2.

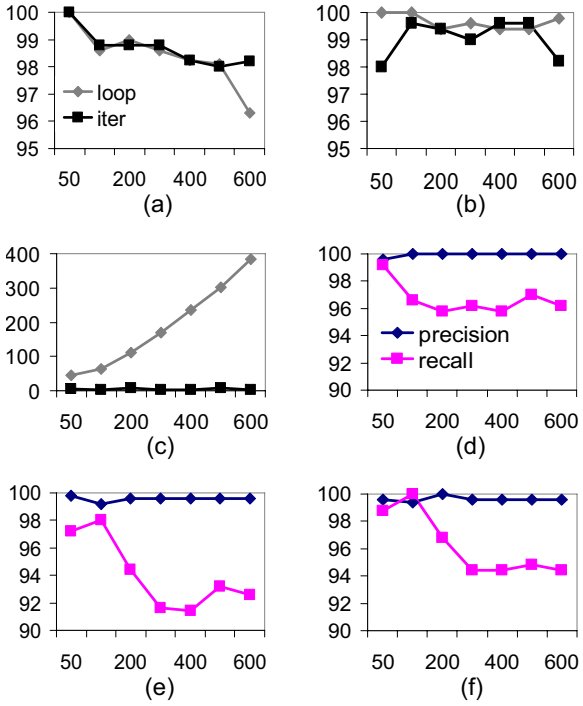

Figure 11: Outlier-robustness. In all diagrams the x-axis represents the number of outliers. The y-axis all diagrams with the exception of diagram (c) represent precision or recall in %. In diagram (c), the y-axis corresponds to the runtime in seconds. Diagram (a)-(c) present a comparison between loopwise and iterative outlier filtering. Diagrams (d)-(f) evaluate the outlier-robustness of the overall OCI-framework. For details see Section 6.2.

call in all experiments). However, iterative repartitioning is much faster than loopwise filtering with a speedup of approximately 100 on the largest data set, as demonstrated in Figure 11(c). Therefore, we apply iterative repartitioning in all experiments. To evaluate the outlier-robustness of the overall OCI framework, we added various amounts of noise points to the whole synthetic data set, starting with the original example with 50 outliers as depicted in Figure 8 up to 40% of outliers. Figures 11(d)-(e) display precision and recall of the identification of the points of cluster 1, 2 and 3. OCI achieved to correctly identify all clusters with precision and recall above 90% in all experiments. In addition, the precision of cluster identification stays close to 100% even for large amounts of outliers.

To cope with the problems caused by outliers in clustering, we could alternatively apply a state-of-the-art outlier detection method prior to clustering e.g. with EM or X-Means. We tried this with LOF [7]. Derived from the principles of density-based clustering, this local outlier factor is in principle suitable to detect outliers w.r.t. clusters of arbitrary shape and object density. LOF detects outliers by comparing the direct object density in the k-neighborhood of a point to the indirect object density in the wider neighborhood of the transitive k-nearest neighbors. As a first drawback, the parameter k has to be specified by the user. But more important, the user also has to select a metric distance function. Moreover, the LOF of each point is continuous value in the range of [0..1] describing to which extend a point is an outlier, thus a suitable cut-off has to be selected if, as in our application, a binary classification into cluster and noise points is required.

Figure 12 depicts the partitioning into cluster points and outliers for $k = 5$ and different cut-off values. As a common cut-off we first marked all points having a LOF greater than two times standard deviation from the mean as outliers, cf. Figure 12(a). With this cut-off we have only few outliers, among them, perhaps surprisingly, some points clearly belonging to the clusters. To remove more outliers, we lowered the cut-off to one time standard deviation, cf. Figure 12(b). Now even more evident, many cluster points are regarded as outliers, and a considerable amount of outliers is still not identified. For a cut-off of one time standard deviation below the mean, most of the outliers are correctly removed as displayed in Figure 12(c). However, most of the cluster points are also classified as outliers. There exists no cut-off value which leads to the correct partitioning into cluster and noise points, due to two reasons: First, the application of LOF leads to problems on complex data distributions with various object densities, as in our example. There is no appropriate global choice of the cut-off value. In contrast, different cut-off values would be appropriate for different regions of the data space. Second, Euclidean distance is no good choice for non-Gaussian data. This effect is most obvious in Figure 12(c) where the clusters are eroded in a Gaussian fashion.

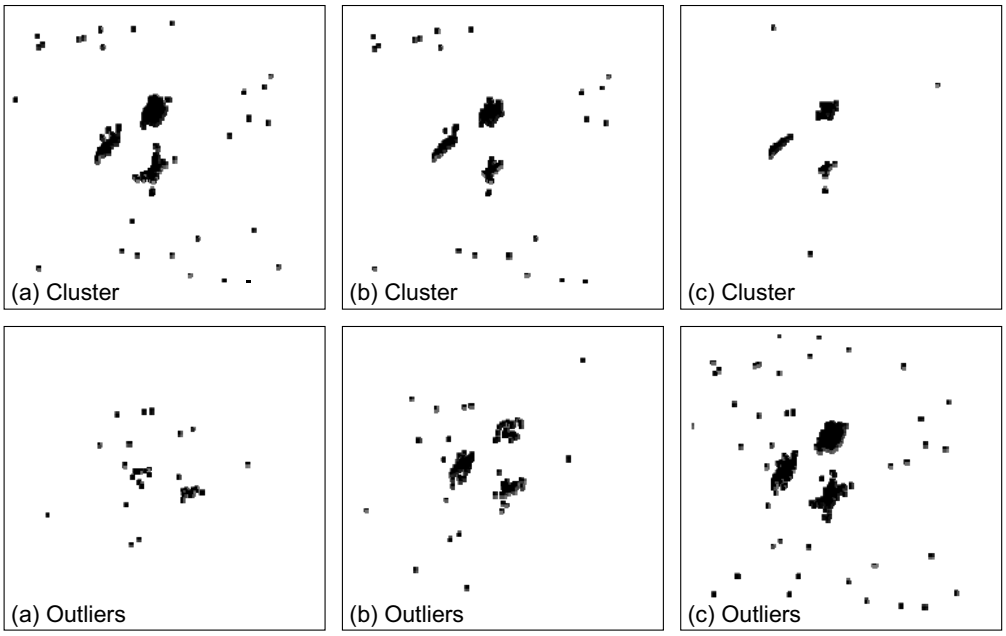

Figure 12: Outlier Detection with LOF. Cluster points and outliers w.r.t. different cut-off values are depicted: (a) outliers are points having a LOF greater than the mean LOF plus two times standard deviation; (b) mean plus one time standard deviation; (c) mean minus one time standard deviation. For details see Section 6.2.

6.3 Effectiveness on Real Word Data

Iris Data. The well known iris data set consists of 150 four dimensional data objects which are labelled to three classes representing different species of the iris plant. As demonstrated in Section 5.1, the clusters are already well separated in the projection on the first splitting direction determined by ICA (cf. Figure 5). The final result of OCI comprises two clusters where one hosts the objects of the class "iris setosa" and the other cluster the objects of classes "iris versicolor" and "iris virginica". The ICA projection in Figure 5 indicates that in fact a hierarchy of three clusters could be detected which would perfectly separate the classes. However, the objects of iris versicolor and iris virginica are very similar, and thus a third cluster does not pay off in coding cost. X-Means and RIC obtain the same result as OCI on iris data. EM also clusters instances of iris versicolor and iris virginica together, where the cross-validation suggests five clusters in total.

Wisconsin Data. The Wisconsin data set deriving from a study on breast cancer consists of 683 instances which are labelled to the classes malign (152 instances) and benign (531 instances) (16 instances with missing values have been removed from the original data set). Each instance is described by nine numerical attributes including e.g. clump thickness, uniformity of cell size and mitoses, which scale between 1 and 10. OCI detects 13 clusters with high class purity on this data set, cf. Table 1. The clusters 1 to 9 are purely composed of benign instances. Most of these clusters exhibit non-Gaussian data distributions and represent different interesting subtypes of benign tumors. By assigning distribution functions to the coordinates, OCI allows a detailed interpretation of the cluster content, which is not possible with the comparison methods. For the 78 instances of cluster 1 e.g. a super-Gaussian distribution with a large beta in all attributes can be observed. The instances of cluster 1 represent tumors with a clump thickness of 3, whereas the uniformity of cell size and shape of most instances is rated by the lowest value of 1. Cluster 8, also super-Gaussian distributed in all attributes, consists of 25 very small tumors with a clump size of 1, which have a slightly enlarged marginal adhesion, and a slightly elevated number of bare nuclei (both values 2 for most instances in this cluster). Cluster 4 comprises objects with sub-Gaussian, almost uniformly distributed clump sizes ranging from 1 to 6, whereas the remaining attributes exhibit a distinct super-Gaussian distribution in the range of 1 to 2 which is not suspicious to cancer. OCI assigns 127 out of 152 instances of the class malign to cluster 13, which is characterized by a highly super-Gaussian distribution of the attribute mitosis which reflects the considerably faster mitosis in cancer tissue, whereas all other attributes exhibit a sub-Gaussian distribution. OCI performs superior regarding the class purity of the clustering which can e.g. be assessed by counting the points assigned to a minority cluster, the so-called minority count, which is 18 (10 benign, 8 maligne) for OCI. The result of EM comprises 5 clusters with a minority count of 35. X-Means and RIC perform better with 3 clusters and a minority count of 32 for X-Means and 6 clusters with a minority count of 20 for RIC. Let us note, that clusterings with different numbers of clusters cannot be directly com-

	cluster-id												
	1	2	3	4	5	6	7	8	9	10	11	12	13
b	78	46	73	41	65	15	29	25	4	94	8	51	2
m	0	0	0	0	0	0	0	0	0	7	17	1	127

Table 1: Classes to Cluster Evaluation of OCI on Wisconsin Data.

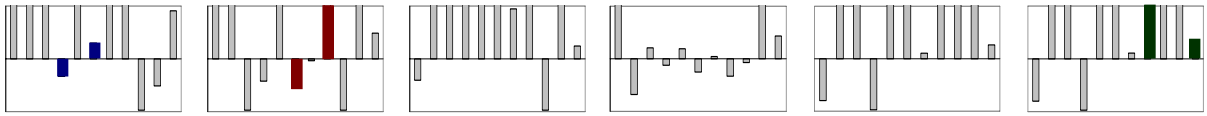

Figure 13: Histograms of the β-values on Housing data. For clusters 1 to 6, the attributes CRIM, ZN, INDUS, NOX, DIS, RAD, TAX, PTRATIO, B and LSTAT are depicted.

pared w.r.t. class purity. However, the different types of benign tumors are best characterized by OCI, because it provides non-Gaussian cluster models.

Housing Data. The housing data set on characteristics of Boston suburbs consists of 506 data objects. The binary valued attribute "CHAS" and the attribute "MEDV" as been left out, resulting in a dimensionality of 11, and the data has been scaled in the range of [0..1]. OCI detects 6 clusters on this data set which exhibit non-Gaussian data distributions. Figure 13 displays for each cluster a histogram of the β values, where every bin corresponds to one attribute. As baseline $\beta = 0$ is selected, which corresponds to a Gaussian distribution and the β values are displayed in the range [0..1]. Features with a super-Gaussian distribution are characterized by a value above the baseline, and features with a sub-Gaussian distribution by a value below the baseline, respectively. It is obvious, that most features in most clusters exhibt distributions which strongly diverge from Gaussian, and also that the pattern of the β distribution is unique for most clusters. Figure 14 displays the projections of some clusters on selected attributes highlighted in Figure 13. For example Figure 14(a) depicts cluster 1 projected on the attributes NOX (nitric oxides concentration) and DIS (weighted distances to five Boston employment centers). In cluster 1, the nitric oxide concentration is moderately sub-Gaussian distributed and the distance to the employment centers exhibts a distribution which is heavy-tailed and therefore best represented by a super-Gaussian distribution. There is no strong correlation between both features in this cluster. Figure 14(b) depicts the projection of the largest cluster 2 comprising 222 objects on the attributes RAD (index of accessibility to radial highways) and PTRATIO (pupil-teacher ratio by town). In cluster 2, the distance to the highways is rather uniformly distributed, where the pupil-teacher ratio has a rather sharp peak around 0.4, resulting in a non-Gaussian cluster of similar shape as our synthetic example depicted in Figure 5(a). In Figure 14(c) the projection of cluster 6 on the attributes PTRATIO and LSTAT (lower state of the population, indicator of whealth) is depicted. This cluster exhibits a sharp peak at PTRATIO of 0.93 and therefore this coordinate can be efficiently compressed using a super-Gaussian distribution with large β, where the lower status is only slightly super-Gaussian. These examples demonstrate that the housing data set contains non-Gaussian clusters with a variety of different data distributions. X-Means detects 4 clusters on this data set, the result of RIC comprises 5 clusters. EM with cross-validation also selects 6 clusters like OCI. However, OCI is the only method providing detailed information about *why* objects are clustered together in terms of a rich vocabulary of distribution functions. The β histograms can be used as unique profiles of the data distribution facilitating the interpretation of the result. Let us consider again cluster 1. From the mean and β histogram e.g. it follows that we have here a cluster of suburbs with very low crime rate, no industry and the pupil teacher ratio is about 0.4, suburbs in cluster 1 can be located at various distance from the working centers.

6.4 Efficiency

Number of Objects. We evaluated the efficiency of OCI and the comparison methods on synthetic data with the same cluster structure as the example displayed in Figure 8 and various numbers of objects starting from 1,550 (the original example) up to 49,600 objects. The results are summarized in Figure 15. Among the comparison methods, X-Means is the fastest algorithm, since no data transformation like PCA or ICA is performed. However, as demonstrated in the previous section, X-Means is not suitable for non-Gaussian data sets with outliers. The EM algorithm with cross-validation is not scalable to large data sets. The algorithm needs more than 10 hours to process the largest data set. Figure 15(ii) provides a close-up of the runtime in the range of 0 to 3,000 seconds. It becomes evident that the runtime of EM with cross-validation is not stable, since it

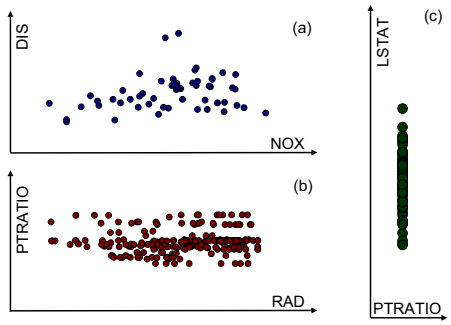

Figure 14: Non-Gaussian Clusters on Housing Data.

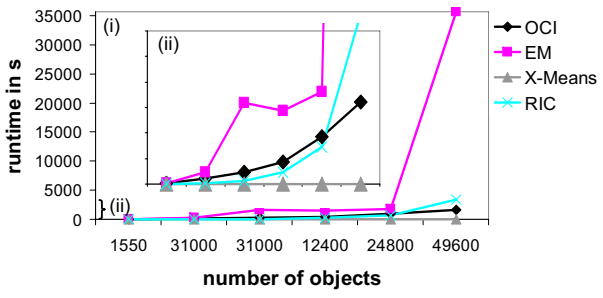

Figure 15: Scalability w.r.t. the Number of Objects.

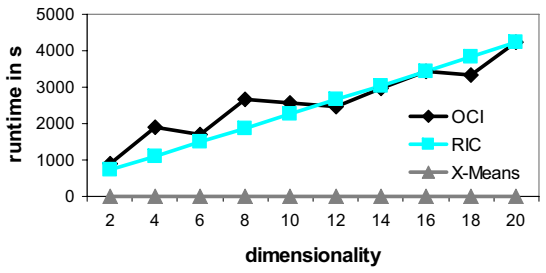

Figure 16: Scalability w.r.t. the Dimensionality.

depends on the random assignment of objects to the folds. OCI scales slightly super-linear with the number of objects and outperforms RIC on data sets larger than approximately 20,000 objects. Providing the most powerful cluster model, OCI is only outperformed by X-Means in efficiency.

Dimensionality. Figure 16 displays the scalability w.r.t. the dimensionality d. Experiments have been performed on a data set with 24,800 objects and varying dimensionality. Not surprisingly, X-Means is the fastest algorithm, since no transformation is performed. RIC and OCI exhibit similar performance which indicates that the PCA is the bottleneck in higher dimensions, and ICA does not cause much additional overhead. The runtime curve of RIC exhibits a direct linear dependency of d, whereas the curve of OCI is not so smooth. This is due to the fact that OCI selects different numbers of clusters on the different data sets, whereas RIC is parameterized with 8 initial clusters in all experiments. EM is left out in this diagram, since, as mentioned, the runtime is very unstable due to cross-validation. Clustering the 4 dimensional data set did not terminate in more than one hour (over 3,600 s). In summary, OCI provides the most powerful cluster model combined with a very good scalability.

7. CONCLUSION

In this paper, we proposed OCI, a novel fully automatic algorithm for clustering non-Gaussian data with outliers. There is a wide variety of clustering approaches but most of them are sensitive to parameter settings and/or heavily affected by single outliers. Many approaches also explicitly or implicitly assume Gaussian data distribution or do not provide a model of the data which is essential in many applications, including selectivity estimation, indexing and classification. OCI has the following desirable properties:

- robust to noise,
- parameter-free,
- uses a flexible model: EPD (Gaussian/uniform/Laplace) combined with ICA.

In an extensive experimental evaluation we demonstrated that OCI successfully detected non-Gaussian clusters in synthetic and real-world data sets. The performance of OCI is stable even in the presence of large amounts of outliers. In addition, OCI is scalable to be used on top of large databases.

8. REPEATABILITY ASSESSMENT RESULT

Figures 8-10 (OCI only) and Figure 15 have been verified by the SIGMOD repeatability committee.

9. REFERENCES

[1] C. C. Aggarwal and P. S. Yu. Finding generalized projected clusters in high dimensional spaces. In *SIGMOD Conference*, pages 70–81, 2000.

[2] R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan. Automatic subspace clustering of high dimensional data for data mining applications. In *SIGMOD Conference*, pages 94–105, 1998.

[3] M. Ankerst, M. M. Breunig, H.-P. Kriegel, and J. Sander. OPTICS: Ordering points to identify the clustering structure. In *SIGMOD Conference*, pages 49–60, 1999.

[4] C. L. Blake and C. J. Merz. *"UCI Repository of machine learning databases, http://www.ics.uci.edu/∼mlearn/MLRepository.html"*.

[5] C. Böhm, C. Faloutsos, J.-Y. Pan, and C. Plant. Robust information-theoretic clustering. In *KDD Conference*, pages 65–75, 2006.

[6] C. Böhm, K. Kailing, P. Kröger, and A. Zimek. Computing clusters of correlation connected objects. In *SIGMOD Conference*, pages 455–466, 2004.

[7] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander. Lof: Identifying density-based local outliers. In *SIGMOD Conference*, pages 93–104, 2000.

[8] A. P. Dempster, N. M. Laird, and D. B. Rubin. "Maximum Likelihood from Incomplete Data via the EM Algorithm". In *J Roy Stat Soc*, number 39, pages 1–31, 1977.

[9] C. H. Q. Ding and X. He. K-means clustering via principal component analysis. In *ICML Conference*, 2004.

[10] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *KDD Conference*, 1996.

[11] P. Grünwald. A tutorial introduction to the minimum description length principle. *Advances in Minimum Description Length: Theory and Applications*, 2005.

[12] G. Hamerly and C. Elkan. Learning the k in k-means. In *NIPS Conference*, 2003.

[13] A. Hyvärinen, J. Karhunen, and E. Oja. *Independent Component Analysis*. 2001.

[14] A. Mineo and M. Ruggieri. A software tool for the exponential power distribution: The normalp package. *Journal of Statistical Software*, 12(4), 1 2005.

[15] A. Ng, M. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm, 2001.

[16] D. Pelleg and A. Moore. X-means: Extending K-means with efficient estimation of the number of clusters. In *ICML Conference*, pages 727–734, 2000.

[17] A. K. Tung, X. Xu, and B. C. Ooi. CURLER: Finding and visualizing nonlinear correlation clusters. In *SIGMOD Conference*, pages 467–478, 2005.

[18] T. Zhang, R. Ramakrishnan, and M. Livny. BIRCH: An efficient data clustering method for very large databases. In *SIGMOD Conference*, pages 103–114, 1996.