International Journal of Data Science and Analytics manuscript No.
(will be inserted by the editor)

Dimension-based Subspace Search for Outlier Detection

Holger Trittenbach · Klemens Böhm

Received: date / Accepted: date

Abstract Scientific data often is high-dimensional. In such data, finding outliers is challenging because they often are hidden in subspaces, i.e., lower-dimensional projections of the data. With recent approaches to outlier mining, the actual detection of outliers is decoupled from the search for subspaces likely to contain outliers. However, finding such sets of subspaces that contain most or even all outliers of the given data set remains an open problem. While previous proposals use per-subspace measures such as correlation in order to quantify the quality of subspaces, we explicitly take the relationship between subspaces into account and propose a dimension-based measure of that quality. Based on it, we formalize the notion of an optimal set of subspaces and propose the *Greedy Maximum Deviation* heuristic to approximate this set. Experiments on comprehensive benchmark data show that our concept is more effective in determining the relevant set of subspaces than approaches which use per-subspace measures.

Keywords Outlier Mining, Subspace Search, High-Dimensional Data

1 Introduction

Outlier detection is a data mining paradigm to discover unusual objects in data. Use cases come from many scientific disciplines where researchers want to identify observations which are exceptional, compared to the bulk of their experimental or simulation data. Regardless of the discipline, the challenge is the same: the detection of rare events, which might even occur for the first time, in large, multi-dimensional data sets.

An object is an outlier if it lies outside of the range of data objects considered normal. To quantify the difference between normal and outlying objects, conventional approaches rely on proximity and locality. However, these principles loose meaning in high-dimensional data spaces, as there is a concentration of distances [23]. The prevalent way of addressing this difficulty is by searching for outliers in low-dimensional projections of the data, where those principles still hold. Subspaces can also increase the interpretability of outliers, by giving them a context where they appear unusual [15,6]. However, the number of subspaces grows exponentially with the data dimensionality. This makes an exhaustive subspace search infeasible. Hence, identifying subspaces where most or all outliers occur is challenging.

Recent approaches detach the search for subspaces from the actual outlier detection [19,18,7]. With this, outlier detection is a two-step process. The first step is to identify subspaces with a high potential to reveal outliers in the subsequent step. Because subspace search is unsupervised, respective approaches must rely on a heuristic measure to quantify this potential. In what follows, we refer to this potential as the subspace quality. In the second step, conventional outlier-detection algorithms, which score the outlierness of each object in a certain space, are applied to the subspaces that are the result of the first step. The scores from each subspace then typically are combined to an overall outlier score per object. This final scoring allows to evaluate the so-called search-result quality, i.e., how well outliers are detected in the set of subspaces. To evaluate

Holger Trittenbach
Karlsruhe Institute of Technology
E-mail: holger.trittenbach@kit.edu

Klemens Böhm
Karlsruhe Institute of Technology
E-mail: klemens.boehm@kit.edu

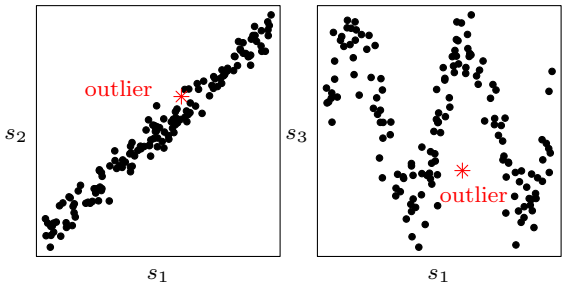

Fig. 1 Two subspaces with different dimensions

the search-result quality, one must compare the outlier scores to an external ground truth.

Since the number of subspaces in a d-dimensional data set is $2^d - 1$, an exhaustive search is intractable. Much work has been devoted to schemes to traverse this search space efficiently. Existing approaches rely on the assumption that a subspace is of high quality if its dimensions are highly correlated. As a result, these so-called correlation-based approaches return a set of highly correlated subspaces. However, it is typically left to the user to pick a cutoff value that discerns "high correlation" from "not-so-high" or some other parameter of this nature.

Example 1 (Motivational example) *Figure 1 depicts two subspaces with different dimensions. The data in the subspace on the left has a stronger correlation than in the right subspace. However, the outlier only occurs in the right subspace, i.e., the combination of Dimensions s_1 and s_3.*

The example shows that, with correlation as the only measure of interest, subspace search may miss subspaces. Additionally, based on current literature [8] and preliminary experiments by ourselves, we hypothesize that adding an irrelevant dimension to a highly correlated subspace reduces the correlation only slightly. In other words, subspaces that also comprise irrelevant dimensions still may have a high correlation. As a consequence, highly correlated dimensions are likely to be overrepresented in the top-k subspaces. Any ranking or thresholding of subspaces purely based on correlation reduces the variety in the final result set.

The discussion so far uncovers a major weakness of existing approaches: Existing measures of subspace quality do not differentiate between the different dimensions of the subspace. This has two implications. First, the subspace-search heuristic might prune all subspaces containing a certain dimension from the search space. Second, any cutoff value to discern between high and low correlation in the search result is arbitrary. The reason is that outliers might only manifest themselves in the dimensions that have been pruned from the search space. So disregarding the subspace dimensions can result in information loss, for these two reasons. The core hypothesis behind this article now is that a more general measure of subspace quality might do away with this problem. Finding such a measure is the problem studied in this paper.

The main contribution of our work is a novel way of assessing subspace quality that goes beyond correlation as the only criterion. We address the weakness of existing approaches in the following way. Instead of assessing the quality of a subspace solely based on its correlation, we propose to assess the quality *per dimension of the subspace*. In Figure 1 for example, the subspace $[s_1, s_2]$ might have a high quality for Dimension s_1 and s_2, while the subspace $[s_1, s_3]$ has a high quality for Dimension s_3. In other words, both subspaces should be included in the result set of subspace search. Comparing subspaces per dimension turns out to be superior to purely correlation-based methods in many scenarios. This is because it allows a differentiated view on the subspaces.

With our dimension-based evaluation, we propose the notion of dominance between sets of subspaces. Intuitively, a subspace S is dominated if there exists a set of other subspaces that have a higher quality for each of the dimensions of S. A dominated subspace should not be in the search result, and we will show that a dominated subspace can even have a negative impact on the outlier detection. With a dominance-based selection, the search objective is to find a non-dominated set of subspaces. This does away with the need for a cutoff threshold. We then propose the *Greedy Maximum Deviation* (GMD), a heuristic that exploits the notion of dominance as part of the subspace search.

We have carried out a comprehensive evaluation of our subspace-search method, with the following criteria:

1. **Search-result quality:** In our context, a set of subspaces is good if it reveals many outliers. We apply an existing outlier-detection method to the subspaces found and use the Area under the Curve (AUC) as the quality measure.
2. **Robustness:** The search-result quality should be insensitive to properties of the data set, such as the proportion of outliers, as well as to the specific parameter values.
3. **Number of subspaces:** Fewer subspaces are better regarding interpretability of the search result.
4. **Runtime:** This is self-explanatory.

According to our evaluation, there is no subspace-search approach which is superior in all scenarios. However, while correlation-based approaches and baseline strate-

gies do return better results than GMD for some of the data sets, GMD outperforms its competitors regarding the above criteria.

Paper outline: After preliminaries in Section 2, we discuss related work in Section 3. In Section 4, we formally refine objectives for a dimension-based subspace search approach. We introduce the Greedy Maximum Deviation heuristic in Section 5 and present experiments in Section 6. Section 7 concludes.

2 Notation

Let DB be a d-dimensional database with a set of dimensions $D = \{s_1, s_2, \ldots, s_d\}$ and N data objects. Each data object x is a vector $\mathbf{x} = (x_{s_1}, x_{s_2}, ..., x_{s_d})$. A subspace S is an orthogonal, lower-dimensional projection of the d-dimensional space. It is a subset of the available dimensions $S \subseteq D$, and we use the notation $S = [s_1, ..., s_d]$ to describe a subspace. Its dimensionality is $|S|$. D is called the full space.

In the following, $RS = \{S_1, S_2, ..., S_k\}$ is a subspace search result. Most of the traditional outlier detection algorithms assign a score value $score(x)$ to each object to quantify its outlierness. Score values can be ordered, and the ranking allows to select the top-k outliers.

The probability density function (pdf) of X is $p(X)$, while we write $\hat{p}(X)$ to refer to the estimation of the pdf. We use $p_{s_i}(X)$ for the marginal pdf of the dimension s_i, i.e., the pdf of the one-dimensional projection on s_i.

3 Related work

3.1 Full space outlier detection

There exists a variety of outlier models from recent years. A common differentiation is by the underlying *outlier definition*, such as cluster-based, distance-based or density-based. However, they all struggle with high dimensionality of data, i.e., the concentration effect in high-dimensional data spaces.

There have been efforts to make full space methods more stable with increasing dimensionality. One example is to use the variance of angles between vectors as a measure of outlierness [12]. Another example for categorical attributes are feature selection methods which remove noisy features from the full space [21,20].

3.2 Subspace search

Subspace search methods examine multiple low-dimensional projections to detect outliers that are hidden in the full space. An exhaustive search is infeasible, because the number of subspaces grows exponentially with the dimensionality of the data. Approaches explicitly searching for interesting subspaces to support the interpretation of outliers are different from approaches like [17,24,22] which rely on random subspace sampling and do not explicitly select subspaces with interesting characteristics. Explicit search approaches fall into two classes: object-based and correlation-based.

3.2.1 Object-based

The idea of object-based subspace search is to identify a set of subspaces for each object. The outlierness is then evaluated based on the object's individual set of subspaces and is often coupled with a specific definition of outlierness [10,11,16,26]. Object-based outlier models can be particularly beneficial for the identification of subspaces where an outlying object deviates most. However, these approaches are prone to overfitting and suffer from the data-snooping bias [28].

3.2.2 Correlation-based

These methods aim to identify one set of subspaces for all data objects. They identify the best set based on the assumption that outliers are more likely to occur in subspaces whose dimension have a high correlation [7,19,18]. Hence, the search for subspaces is decoupled from the actual outlier model. Existing purely correlation-based approaches differ in the way they estimate the correlation in subspaces and how they traverse the search space.

The decoupling of subspace search from outlier detection was first introduced in [7]. The authors proposed HiCS, which quantifies the correlation in subspaces as the difference between the expected joint distribution, assuming independence between dimensions, and the observed data distribution. The difference is called the *contrast of a subspace*.

$$contrast_{HiCS}(S) = diff(p^S_{expected}, p^S_{observed}) \quad (1)$$

The contrast is 0 for

$$p^S_{expected} = p^S_{observed} = \prod_{i=1}^{d} p_{s_i}(X_i) \quad (2)$$

with $S = \{s_1, ..., s_d\}$. The more the conditional pdf deviates from the marginal pdf, the more the independence assumption is violated. Estimating the observed joint pdf for a multi-dimensional subspace is not trivial. Therefore, HiCS uses a dynamic slicing concept to avoid the problem of data sparsity in high-dimensional spaces. A *subspace slice* basically is an adaptive grid

cell in the higher-dimensional space. It is constructed for a dimension $s_i \in S$ by restricting the domain of all *remaining dimensions* $s_j \in S$, $s_j \neq s_i$ to an interval $[l_j, r_j]$, $l_j \leq r_j$. We refer to the set of conditions for the reference attribute s_i as C_{-i}. The pdf for the reference attribute on the subspace slice is the *conditional pdf*.

$$p_{s_i|C_{-i}} = p(x_i \,|\, x_j \in c_j \;\forall\; j \in \{1..|S|\} \setminus i) \tag{3}$$

The idea is that, if a subspace is uncorrelated, restricting the data on a subspace slice C_{-i} for any $s_i \in S$ does not change the distribution of s_i. I.e., the conditional pdf is the same as the marginal pdf for all possible reference attributes of a subspace. This makes sense intuitively, because it means that all dimensions are mutually independent. For mutually independent dimensions, the joint pdf $p_{s_1,\ldots,s_d}(X)$ is the product of all marginal pdfs $p_{s_i}(X)$. The *deviation* function, or short *dev*, quantifies the violation of the independence. It assesses how much the conditional pdf deviates from the marginal pdf. The difference in Equation 1 is the average deviation of the conditional pdf and marginal pdf over all attributes $s_i \in S$.

$$contrast_{HiCS}(S) = \frac{1}{M} \sum_{m=1}^{M} dev(\hat{p}_{s_i}, \hat{p}_{s_i|C_{-i}}) \tag{4}$$

It is approximated by a Monte Carlo simulation with M iterations and a random selection of the reference attribute s_i and subspace slice C_{-i} in each iteration m.

One must select an appropriate deviation function. In this work we stick to the two-sample Kolmogorov-Smirnov (KS) test. It has been used in the benchmark experiments of [7]. The test statistic uses the peak of the difference between the marginal and conditional cumulative distribution functions F.

$$dev(\hat{p}_{s_i}, \hat{p}_{s_i|C_{-i}}) = \sup_{x_i \in DB} |\hat{F}_{s_i}(x_i) - \hat{F}_{s_i|C_{-i}}(x_i)| \tag{5}$$

The larger the peak difference, the higher the deviation. Note that the deviation function using the KS test is bound to the interval $[0, 1]$.

HiCS follows an Apriori-style search scheme to identify the set of the highest contrast subspaces. Other approaches like 4S [18] and CMI [19] have been proposed to overcome the shortcomings of Apriori. They all introduce their own notion of correlation. However, all these approaches define the subspace quality solely based on the correlation, regardless of the subspace dimensions.

In this work, we focus on correlation-based methods. We use the well-known LOF algorithm [3] as the outlier detection method in the subspaces, as it is widely used as a reference point [7,18,19].

3.3 Score combination

For correlation-based methods, an outlier algorithm is applied to each subspace, and the final score is calculated by combining the individual scores. The most prominent combination functions have been maximum and average. The right choice of combination function is application dependent and has been discussed controversially in literature [1,27]. In this work we rely on the summation of scores.

$$score_{final}(x) = \sum_{S \in RS} score_S(x) \tag{6}$$

4 Subspace search objectives

Subspace search aims at identifying the subspaces that contain outliers. In general, outliers do not form a homogeneous class in the data, i.e., they do not follow a common distribution. Consequently, one cannot directly assign a probability to a subspace that indicates how likely it contains outliers. Instead, we say that a subspace has a *potential to reveal outliers* if there are areas where non-trivial outliers [9] can be placed. This potential of a subspace depends on the underlying type of outlier definition. Under the density-based definition, we consider a subspace to have such a potential if projections of low-density areas fall into high-density areas in the low-dimensional subspaces.

Our goal is to derive a search objective that is independent of a specific outlier detection algorithm. We discuss the search objective of purely correlation-based methods in Section 4.1. We then examine the identified weakness of correlation-based methods in some detail before we propose our novel, dimension-based approach in Section 4.3.

4.1 Weakness of correlation-based approaches

We now discuss the weakness of correlation-based approaches. The search objective of these approaches is to identify the top-k subspaces with the highest correlation. Existing algorithms differ in the way they construct the candidates, but their focus is on maximizing the correlation between attributes.

Example 2 (Discarded subspace) *Think of a subspace $[s_1, s_2]$ which shows a slightly higher correlation than the subspace $[s_2, s_3]$. During the search, the subspace $[s_2, s_3]$ might be pruned because it is not one of the best subspaces that are kept during the search. Although we have not formally introduced dominance yet, one might agree intuitively that this is because of the*

assumption that $[s_1, s_2]$ *dominates* $[s_2, s_3]$*. In consequence, subspace* $[s_2, s_3]$ *is discarded in favor of* $[s_1, s_2]$*. If no other subspaces that contain Dimension* s_3 *are in the final result set, then there is no way to find any outlier that manifests itself in subspaces with Dimension* s_3*.*

So current approaches rely only on the correlation measure, and do not consider the diversity of the result set. Also, it is not possible to come up with any statement on how many subspaces need to be searched for outliers, or which correlation values are deemed good enough for subsequent consideration. The prevalent way to bypass this issue is to use only the top-k subspaces identified as input to the outlier detection.

Furthermore, it is problematic to assess two different sets of subspaces simply by the correlation of the subspaces they consist of. The reason is that highly correlated dimensions can quickly be overrepresented in the result set. This is particularly true if the subspace search algorithm follows an elitist approach and only keeps track of the best subspaces found.

Example 3 (Inflated search result) *Table 1 illustrates the result set produced by HiCS on one of its benchmarking data sets. Dimensions* s_{19} *and* s_{20} *are part of all but three subspaces among the top-20 results. This is a result of* $[s_{19}, s_{20}]$ *having a high correlation, and adding an additional, non-correlated dimension, only affects the correlation slightly. The diversity among the top-20 subspaces is low.*

Rank	HiCS contrast	Subspace
1	0.47	$[s_{19}, s_{20}]$
2	0.34	$[s_{12}, s_{19}, s_{20}]$
3	0.34	$[s_9, s_{10}, s_{11}, s_{12}, s_{13}]$
4	0.33	$[s_6, s_{19}, s_{20}]$
5	0.33	$[s_9, s_{19}, s_{20}]$
6	0.33	$[s_1, s_2, s_3]$
7	0.32	$[s_{11}, s_{19}, s_{20}]$
8	0.32	$[s_3, s_{19}, s_{20}]$
9	0.32	$[s_1, s_{19}, s_{20}]$
10	0.32	$[s_{14}, s_{15}, s_{16}, s_{17}, s_{18}]$
11	0.31	$[s_{10}, s_{19}, s_{20}]$
12	0.31	$[s_{15}, s_{19}, s_{20}]$
13	0.31	$[s_{14}, s_{19}, s_{20}]$
14	0.31	$[s_5, s_{19}, s_{20}]$
15	0.30	$[s_7, s_{19}, s_{20}]$
16	0.29	$[s_{17}, s_{19}, s_{20}]$
17	0.29	$[s_2, s_{19}, s_{20}]$
18	0.29	$[s_{16}, s_{19}, s_{20}]$
19	0.29	$[s_{18}, s_{19}, s_{20}]$
20	0.29	$[s_{13}, s_{19}, s_{20}]$

Table 1 Results from the HiCS search on a 20-dimensional synthetic benchmark data set

Consequently, a meaningful comparison between two result sets is only possible ex-post, i.e., based on the outlier scorings. This requires to run an outlier detection algorithm on each result set and to match the final scores against a ground truth. However, this contradicts the nature of an unsupervised search where no information on the object class labels is available.

4.2 Symmetry in other correlation measures

The loss of information by averaging the deviations over all dimensions of the subspace is not unique for HiCS. Any subspace search approach that uses a symmetric correlation measure on a subspace suffers from this weakness. The reason mainly is that other statistical correlation measures use joint information on the variables instead of combining one-dimensional information. We exemplarily examine Mutual Information (MI), Spearman Correlation and the Cumulative Mutual Information (CMI) [19] regarding their symmetry.

MI is one of the most commonly used measures to describe the relationship between two variables. It is defined as

$$I(X;Y) = \iint p(x,y)\ log\frac{p(x,y)}{p(x)p(y)}\ dydx \tag{7}$$

It describes the dependency of two random variables and can be used as a correlation measure. Since the joint probability function $p(x, y)$ and the product of the marginal distributions $p(x)p(y)$ are symmetric, the measure itself is symmetric.

$$I(X;Y) = I(Y;X) \tag{8}$$

The Spearman correlation coefficient quantifies the relationship between the ranks of ordered variables. It is based on the covariance between ranks, which is symmetric. Therefore, the coefficient is symmetric as well.

CMI is a more involved approach to estimate correlation in multi-dimensional spaces. The CMI of a subspace consisting of the random variables $X_1, ..., X_d$ is

$$CMI_{[X_1,...,X_d]} = \sum_{i=2}^{d} diff\,(p(x_i), p(x_i|x_1, ..., x_{i-1})) \tag{9}$$

The idea is to sum up the contrast of the lower-dimensional projections $(X_1, X_2), (X_1, ..., X_i), ..., (X_1, ..., X_d)$ of the subspace. Hence, the CMI depends on the permutation of the dimensions of the subspace. However, the contrast of a subspace is defined as the maximum CMI over all permutations π.

$$contrast_{CMI}(S) = CMI(\pi_{opt}(X_1, ...X_d)) \tag{10}$$

with π_{opt} as the permutation with maximal CMI. Thus, the contrast is symmetric for all dimensions as well.

4.3 Dimension-based approach

Our goal is to derive a notion of optimality of a set of subspaces. Intuitively, a set of subspaces is optimal if adding or removing subspaces will affect the search-result quality, i.e., how well outliers are detected within the set, negatively. We first introduce some definitions to describe the optimality of a search result formally. Then, we propose a framework that allows to compare different sets of subspaces based on individual dimensions. We show that our framework finds an optimal set of subspaces without the need for any cutoff value.

Definition 1 (Superfluousness of a subspace) A subspace S is *superfluous* if it can be excluded from a set of subspaces without a negative effect on the results of the subsequent outlier detection.

Even though there may be a set of subspaces where none of the subspaces is superfluous, there might be another set that consists of fewer subspaces and is of the same quality. This gives way to the following definition.

Definition 2 (Minimal set of subspaces) A set of subspaces is *minimal* if no subspace is superfluous, and a union of two or more of the subspaces has a negative effect on the search-result quality.

Example 4 (Minimal set) *Consider a database* $D = \{s_1,\ s_2, s_3\ s_4, s_5\}$ *and the set* $S_1 = \{[s_1, s_2],\ [s_1, s_3],\ [s_4, s_5]\}$. *None of the three subspaces in* S_1 *is superfluous. The reason is that removing any of the subspaces in* S_1 *would exclude one of the dimensions* s_2 *or* s_3 *or both* s_4 *and* s_5 *from the set. However, according to Definition 1 the set* $S_2 = \{[s_1, s_2, s_3], [s_4, s_5]\}$ *is preferred over* S_1 *if* S_2 *yields the same or a better search-result quality.* S_2 *is also minimal according to Definition 2 if the search-result quality on the set* $S_3 = \{[s_1, s_2, s_3, s_4, s_5]\}$ *is lower than on* S_2.

We now propose the following optimality criterion.

Definition 3 (Optimal set of subspaces) Given an outlier definition *OutDef*, a set of subspaces *RS* is *optimal* if

1. there exists no other set of subspaces $RS' \neq RS$ that increases the search-result quality for an outlier model based on *OutDef*, and
2. RS is minimal.

RS abstracts from specific outlier models and only relies on the underlying outlier definition (e.g., distance-based or density-based). Because of the broad variety of meaningful outlier models, there are different instantiations of *OutDef* with different search-result quality. However, changing *RS* cannot improve this quality for any of these instantiations. The evaluation of the search-result quality depends on the metric used. A common metric is the area under the ROC curve.

To sum up, using the optimal set of subspaces as a search objective is reasonable, but bears difficulties in practice. In particular, the search-result quality depends on an ex-post evaluation that is bound to an outlier detection algorithm. Thus, in unsupervised subspace search, one cannot make use of the optimality criterion directly.

4.4 Framework for subspace comparison

In an unsupervised setting, the potential of subspaces to reveal outliers needs to be comparable without an ex-post evaluation. To achieve this, we link the quality of a subspace to its dimensions.

Definition 4 (Subspace-Quality Function) Let S be a subspace and $s_k \in D$ be a dimension of the full space D. A *Subspace-Quality Function (SQF)* is a function of type

$$q : D \times \mathcal{P}(D) \rightarrow [0, 1]$$

with the property

$$q(s_k, S) = 0 \quad \forall s_k \notin S.$$

The function q is a measure of subspace quality. Because an SQF is dimension-specific, a subspace with d dimensions corresponds to d different SQF values. Intuitively, $q(s_k, S)$ is zero if Dimension s_k is not part of the subspace, i.e., $s_k \notin S$. The reason is that the subspace cannot reveal outliers for s_k if this dimension is not part of the subspace. We further require that SQF values need to be *directly comparable* for the same reference dimension.

Definition 5 (Directly comparable SQF) Let two subspaces S, S' be given that both contain a dimension s_k. Then an SQF q is directly comparable with regard to s_k if

$$q(s_k, S) > q(s_k, S'), S \neq S'$$

means that S has a higher potential to reveal outliers for s_k.

It is straightforward to compare two subspaces by their correlation. However, this comparison becomes slightly more involved with SQF values. We first define a partial ordering:

$$S_A \succeq_q S_B \iff \forall s_k \in S_A \cup S_B : q(s_k, S_A) \geq q(s_k, S_B) \quad (11)$$

We illustrate why this order is partial with an example.

Example 5 (Partial ordering) *Consider the two subspaces* $S_A = [s_1, s_2, s_4]$ *and* $S_B = [s_1, s_2, s_3]$ *with*

$q(s_1, S_A) = 0.6,\ q(s_2, S_A) = 0.7$ *and* $q(s_4, S_A) = 0.2$

$q(s_1, S_B) = 0.5,\ q(s_2, S_B) = 0.6$ *and* $q(s_3, S_B) = 0.7.$

With Equation 11, we observe $S_A \not\succeq_q S_B$ *and* $S_A \not\preceq_q S_B$ *because* $q(s_3, S_A){=}0$ *and* $q(s_4, S_B){=}0$.

Consequently, this comparison is only useful when comparing subsets of subspaces. So the usefulness of Equation 11 is limited. Therefore, we move away from comparing subspaces with each other and compare a subspace to a set of subspaces.

Definition 6 (Set of subspaces $\succ$ subspace) Let $\mathcal{S}$ be a set of subspaces where each subspace is a subset of the full space D, and let q be an SQF. Let S_A be a subspace with $S_A \subseteq D,\ S_A \notin \mathcal{S}$. Then $\mathcal{S}$ *dominates* $\boldsymbol{S_A}$ with respect to q if the following holds:

$$\mathcal{S} \succ_q S_A \quad \Leftrightarrow \quad \forall s_k \in S_A : \exists S' \in \mathcal{S} \quad s.t. \quad q(s_k, S') > q(s_k, S_A) \tag{12}$$

A set of subspaces $\mathcal{S}$ dominates a subspace S_A if for every dimension of S_A there is at least one subspace $S' \in \mathcal{S}$ with a higher SQF value. The rationale is to rule out compensation of SQF values across dimensions.

So far, we have defined domination only in one direction, such that a set of subspaces can dominate a single subspace. The inversion also holds. This is because a single subspace can dominate a set of subspaces as well. This inverted relation is also true with equality, as a single subspace should be preferred over a set of subspaces. We hence complement Definition 6 with

Definition 7 (Subspace $\succ$ set of subspaces) A *subspace* S_A *dominates a set of subspaces* $\mathcal{S}$:

$$S_A \succ_q \mathcal{S} \quad \Leftrightarrow \quad \forall s_k \in S_A, \forall S' \in \mathcal{S} : \quad q(s_k, S') \leq q(s_k, S_A) \tag{13}$$

This means that a subspace can replace other subspaces in the current solution if it dominates them.

Example 6 (Subspace Dominance) *Consider subspaces* $S_A = [s_1, s_2, s_3],\ S_B = [s_1, s_3]$ *and* $S_C = [s_2, s_3]$ *with*

$q(s_1, S_A) = 0.5,\ q(s_2, S_A) = 0.6,\ q(s_3, S_A) = 0.7$

$q(s_1, S_B) = 0.4,\ q(s_3, S_B) = 0.5$

$q(s_2, S_C) = 0.6,\ q(s_3, S_C) = 0.6$

Then $S_A \succ_q \{S_B, S_C\}$.

With this new notion of dominance, subspaces are directly comparable.

From Definition 6 and Definition 7, one can derive a set of subspaces where no subspace is dominated.

Definition 8 (Non-dominated set of subspaces) A *non-dominated set of subspaces* is a set of subspaces $\mathcal{S}_{ND}$ where

$$\forall \mathcal{S}' \subseteq \mathcal{S}_{ND},\ \forall S_A \in \mathcal{S}_{ND},\ S_A \notin \mathcal{S}' : \quad \mathcal{S}' \not\succ S_A \wedge S_A \not\succ \mathcal{S}' \tag{14}$$

Within a set of non-dominated subspaces $\mathcal{S}_{ND}$, it is not possible to find a subset $\mathcal{S}'$ and a subspace $S_A \notin \mathcal{S}'$ such that the subset dominates the subspace or vice versa. We cannot remove any subspace from the solution without risking a negative effect on the results of the subsequent outlier detection. None of the subspaces in a non-dominated set is superfluous.

Instead of searching the top-k subspaces, it is possible to apply the domination principle to find a non-dominated set of subspaces in the following way. An SQF evaluates the subspaces encountered during the search. For each new subspace, we can apply Definition 6 to check if the current solution dominates the subspace found. If so, the subspace can be discarded. Otherwise, it extends the non-dominated set.

Example 7 (Motivational example cont.) *Let us assume that the search encounters the subspace* $[s_1, s_2]$ *as the first subspace, i.e.,* $RS = \emptyset \cup [s_1, s_2]$. *In the next step, the search encounters* $[s_1, s_3]$. *By Definition 6, it is* $[s_1, s_2] \not\succ_q [s_1, s_3]$ *and* $[s_1, s_2] \not\prec_q [s_1, s_3]$. *Hence,* $[s_1, s_3]$ *extends the non-dominated set to* $RS = \{[s_1, s_2], [s_1, s_3]\}$.

We now show that one can find an optimal set of subspaces by using this strategy. For each dimension in the full space, it is possible to find a subspace S with the maximum SQF value for that dimension, i.e.,

$$q(s_k, S) > q(s_k, S') \quad \forall S' \subseteq D,\ S \neq S'$$

We obtain an optimal set of subspaces with regard to the SQF by identifying such a subspace for each dimension.

Theorem 1 (SQF and optimality) *If q is an outlier identifying SQF and D a set of dimensions, then a non-dominated set*

$$OS = \{\mathcal{S} \subseteq \mathcal{P}(D) \mid \forall s_k \in D\ \exists S \in \mathcal{S}\ \forall S' \subseteq D : \quad q(s_k, S) > q(s_k, S')\ \} \tag{15}$$

is an optimal set of subspaces with regard to q.

Proof Theorem 1 satisfies the first axiom of Definition 3. OS is defined such that, for each dimension in D, there exists a subspace in OS which maximizes the SQF. Higher values of an outlier-identifying SQF stand for a higher potential of identifying outliers. For an SQF with this characteristic, the best set that one can find contains a subspace for each dimension that maximizes the SQF.

OS does not contain superfluous subspaces. Any subspace that does not have the maximal SQF value for at least one dimension is dominated by OS. Because OS is non-dominated, such a subspace cannot exist in OS. If we could replace two subspaces with one, this single subspace would need to have a higher SQF value for each dimension. Such a subspace does not exist either. This is because every subspace in OS already maximizes the SQF for one dimension. Hence, OS is minimal and also satisfies the second axiom of Definition 3.

In general, it is also possible to retain multiple non-dominated sets during the search. One would add a subspace detected to all the sets where it is non-dominated. Using our framework to find several, alternate solutions is future work.

5 Maximum deviation subspaces

So far, we have not proposed any concrete instantiation of the SQF. In the following we instantiate the SQF with a deviation function which has been used for the purely correlation-based subspace search in [7]. We first generalize the deviation function so that it becomes an SQF and then propose the Greedy Maximum Deviation (GMD). GMD is a dimension-based subspace-search approach, a heuristic to approximate the optimal set of subspaces.

5.1 Deviation function as an SQF

We observe that one can generalize the definition of deviation in Equation 5 to an SQF as follows.

$$dev(s_k, S) = \begin{cases} 0 & \text{if} s_k \notin S \\ dev(\hat{p}_{s_k}, \hat{p}_{s_k|C_{-k}}) & \text{otherwise} \end{cases} \quad (16)$$

Instead of averaging deviations over all dimensions of a subspace to calculate the contrast, as in Equation 4, the deviation can be calculated individually for each Dimension s_k of the subspace. $dev(s_k, S)$ fulfills the properties of a directly comparable SQF according to Definitions 4 and 5. The image of $dev(s_k, S)$ is $[0, 1]$ if we use the KS test as an instantiation of the deviation function (see Equation 5). $dev(s_i, S)$ and $dev(s_i, S')$ also are directly comparable: A higher deviation means that, on average, the difference between the marginal and conditional distribution is higher. Hence, there are more areas in the data space where non-trivial outliers can occur, i.e., the potential to reveal outliers is higher.

With the framework introduced in Section 4.4 and with Definition 16, we can return to Example 3 and offer an explanation why the search result is inflated.

Example 8 (Inflated search result cont.)
The reason for the inflated search result becomes obvious when looking at the individual deviations, instead of the overall contrast. For the subspace $[s_2, s_{19}, s_{20}]$ *at rank 17 in Table 1, we calculate the deviation of the individual dimensions according to Equation 16.*

$$dev(s_2, [s_2, s_{19}, s_{20}]) = 0.15$$
$$dev(s_{19}, [s_2, s_{19}, s_{20}]) = 0.37$$
$$dev(s_{20}, [s_2, s_{19}, s_{20}]) = 0.39$$

Dimension s_2 *has a small deviation compared to* s_{19} *and* s_{20}*. But if we take the subspace at rank 6, we observe a better subspace with* $dev(s_2, [s_1, s_2, s_3]) = 0.36$*. We also observe*

$$dev(s_{19}, [s_{19}, s_{20}]) = 0.45$$
$$dev(s_{20}, [s_{19}, s_{20}]) = 0.46$$

Hence, according to Definition 6, we infer the relationship

$$\{[s_1, s_2, s_3], [s_{19}, s_{20}]\} \succ_{dev} [s_2, s_{19}, s_{20}]$$

We can therefore exclude the subspace $[s_2, s_{19}, s_{20}]$ *from the result set. Because the contrast is an average over all the dimensions, other dimensions can be added to* $[s_{19}, s_{20}]$ *with only a minor decrease of the overall contrast. This explains the overrepresentation of* $\{s_{19}, s_{20}\}$ *in the top-20 subspaces.*

Evaluation of the initial result, the top-20 subspaces found by HiCS, by using LOF on each subspace and by summing up the scores yields an AUC of 0.80*. If we exclude the superfluous subspaces, the remaining set is* $\{[s_{19}, s_{20}]$, $[s_9, s_{10}, s_{11}, s_{12}, s_{13}]$, $[s_1, s_2, s_3]$, $[s_{14}, s_{15}, s_{16}, s_{17}, s_{18}]\}$ *with* $AUC = 0.84$*. This means that we can improve the search-result quality by removing 16 out of the top-20 subspaces. This is because they reduce the ability to correctly discern outliers from regular objects.*

5.2 Greedy maximum deviation

To identify the optimal set of subspaces according to Theorem 1, one needs to find the subspace maximizing

Algorithm 1 Greedy Maximum Deviation (GMD)

Input: Data set D with dimensions $\{s_1, s_2, ..., s_d\}$
Output: Set of subspaces RS with one subspace per dimension in D

1: $RS \leftarrow \emptyset$
2: **for** s_i in D **do**
3: $SC \leftarrow \{[s_i, s_j] \mid s_j \in D \backslash \{s_i\}\}$
4: $T_{s_i}^{max} \leftarrow \arg\max_{T \in SC} dev(s_i, T)$
5: $SC \leftarrow SC \setminus \{T_{s_i}^{max}\}$
6: **while** $|SC| > 0$ **do**
7: $T_{cand} \leftarrow \arg\max_{T \in SC} dev(s_i, T)$
8: **if** $dev(s_i, T_{s_i}^{max} \cup T_{cand}) > dev(s_i, T_{s_i}^{max})$ **then**
9: $T_{s_i}^{max} \leftarrow T_{s_i}^{max} \cup T_{cand}$
10: **end if**
11: $SC \leftarrow SC \setminus T_{cand}$
12: **end while**
13: $RS \leftarrow RS \cup T_{s_i}^{max}$
14: **end for**

the deviation for each dimension. Due to the exponential number of subspaces, a brute force search is not feasible. Instead, we propose Greedy Maximum Deviation (GMD), a heuristic to approximate the optimal set of subspaces.

GMD is a dimension-based heuristic, because it uses a SQF to quantify the subspace quality per dimension. During the search, GMD constructs subspaces in a greedy way. In each search step, it picks the solution which improves the objective function the most. The idea is that we iterate over each dimension $s_i \in D$ and construct a subspace that strives to maximize the deviation for this dimension. This leads to an output of $|D|$ subspaces that form a non-dominated set. Because this is the only search criterion, the dimensionality of the maximum deviation subspaces can differ for the dimensions in D.

The pseudo code for GMD is in Algorithm 1. In the following, s_i is the reference attribute and $T = [s_i, s_j], i \neq j$ a two-dimensional subspace. $dev(s_i, T)$ is the deviation for the reference attribute s_i in the subspace T. SC is a set of two-dimensional subspaces. SC is initialized with the Cartesian product of all dimensions with the reference attribute s_i (Line 3). The initial subspace $T_{s_i}^{max}$ is the two-dimensional space with the highest deviation for the reference dimension s_i (Line 4). In each iteration, the subspace T_{cand} with the highest deviation among the remaining two-dimensional projections is chosen. If the reference dimension and the added dimension show a high deviation in the two-dimensional projection, we hypothesize that this relation also has an impact on the deviation in higher-dimensional subspaces. This monotonicity assumption is weaker than the one behind HiCS. There, every lower-dimensional projection of a subspace needs to have a high contrast.

If the deviation for s_i in the subspace $T_{s_i}^{max} \cup T_{cand}$ increases, the best solution found so far is updated (Line 9). T_{cand} is then removed from the set of remaining subspaces. Unlike HiCS, there is no need for a subspace to have a *high* contrast. Instead, *any* positive increase of the deviation leads to the inclusion of the dimension in the subspace. The iteration stops if SC is empty, i.e., there are no two-dimensional subspaces left to consider. The final subspace for each dimension is added to the result set (Line 13).

Note that the result of GMD is independent of the ordering of D. This is because the heuristic constructs a subspace individually for each dimension, independent of the subspaces of the other dimensions. The two-dimensional deviations determine the order in which a dimension is added to $T_{s_i}^{max}$.

Example 9 (GMD subspace construction)
Think of a five-dimensional full space, and we apply GMD to it. GMD iterates over all subspace dimensions and starts with s_0. Table 2 illustrates the subspace construction. The two-dimensional subspaces for s_0 are ordered by decreasing $dev(s_0, [s_0, s_j]: j \neq 0)$. Then, step by step, GMD adds dimensions s_1, s_4, s_3, and s_2 and calculates the deviation of the extended subspace. In the second step, GMD discards s_3 because adding s_3 to the subspace does not improve the deviation for s_0.

5.3 Runtime complexity

The runtime complexity of outlier-detection approaches that include subspace search depends on two variables: the complexity of the subspace search algorithm and the number of subspaces in the result set that are searched for outliers.

5.3.1 Complexity of GMD

The runtime complexity of GMD depends on the one of two subproblems, referred to as *search* and *index*. We use $C(\cdot)$ to refer to the runtime complexity of the subproblems.

$$C(GMD) = C(search) + C(index) \tag{17}$$

We first derive $C(index)$. The index keeps the order of the data objects for each dimension individually. Hence, building the index has the complexity of sorting the N data objects in each dimension.

$$C(index) \in O(|D| \cdot N \cdot log(N)) \tag{18}$$

We now derive $C(search)$. It depends on the number of subspaces the deviation is calculated for and

s_j	$dev(s_0, [s_0, s_j])$
s_1	0.4
s_4	0.35
s_3	0.28
s_2	0.27

step	subspace S	$dev(s_0, S)$
0	$[s_0, s_1]$	0.4
1	$[s_0, s_1, s_4]$	0.42
2	~~$[s_0, s_1, s_3, s_4]$~~	~~0.35~~
3	$[s_0, s_1, s_2, s_4]$	0.49

Table 2 Example for GMD subspace construction

on the complexity of this calculation. For each dimension, GMD examines all two-dimensional projections (Algorithm 1, Lines 4–5). Then GMD selects the two-dimensional projection with the highest deviation and adds each of the other dimensions step by step (Lines 6–12).

The selection of the highest deviation needs a calculation of $|D| - 1$ two-dimensional deviations and a sort of the result set. The step-by-step addition results in another $|D| - 2$ deviation calculations. So the total number of deviation calculations is $|D| - 1 + |D| - 2$. Consequently, the complexity of the step-by-step addition is in $O(|D| \cdot C(dev))$. Sorting the two-dimensional projections is in $O(|D| \cdot \log(|D|)$. GMD repeats these calculations for all dimensions in the data set, i.e., $|D|$ times (Line 2). Hence, the resulting complexity is

$$C(search) \in O\left(|D| \cdot [|D| \cdot \log |D| + |D| \cdot C(dev)]\right) \quad (19)$$

Next, we derive $C(dev)$, the complexity of the deviation calculation. The deviation is calculated by the KS test on a random subspace slice. Let the result of the slicing be a vector $r = (r_1, r_2, ..., r_N)$ of N bits, where r_i with $i = 1, \ldots, N$ relates to the i-th object of the data set. At the beginning of the slicing, r is initialized with 1. Then the slicing procedure selects a random interval between 0 and N for each of the non-reference dimensions. For each of these intervals, the values of r outside of the interval are set to 0. After the slicing, the bits with value 1 indicate the objects that remain in the slice.

The initialization of r is in $O(N)$. The selection of the random interval can be performed in $O(1)$. This is because the index stores the order of the objects for each dimension individually. The random interval in one dimension is just a block of consecutive entries in the index. So selecting an interval reduces to a random selection of one of the interval borders. The number of objects outside of the interval, for which the bit in r is set to 0, depends on the parameter $\alpha \in [0, 1]$ and is $(1 - \sqrt[|S|-1]{\alpha}) \cdot N$ [7]. With the number of dimensions in a subspace $|S|$, the number of interval selections required to extract a subspace slice is $|S| - 1$. Hence, the slicing is in $O\left(|S| \cdot N\right)$.

The KS test statistic (cf. Equation 5) relies on the pointwise distance between the cumulative distribution function of the conditional and of the marginal distribution. One can calculate it directly on r.

$$KS = \sup_{k \in \{1,\ldots,N\}} \left| \frac{k}{N} - \sum_{i=1}^{k} \frac{r_i}{\sum_{j=1}^{N} r_j} \right| \quad (20)$$

The test statistic is in $O(N)$ because it needs one pass over r. The deviation is then averaged over M Monte Carlo iteration results. Hence, the resulting complexity is

$$C(dev) \in O\left(N + (|S| \cdot N) \cdot M\right) \quad (21)$$

In our scenario, α and M are external parameters and are constant for all our experiments. In addition, subspace search is effective on samples of the data. Therefore, we subsample to a constant number of objects, i.e., N is a constant. In this case, (21) and (18) reduce to

$$C(dev) \in O(|S|) \quad (22)$$

$$C(index) \in O(|D|) \quad (23)$$

Substituting (22) into (19) and (23) and (19) into (17) gives

$$C(GMD) \in O(|D|^2 \cdot log(|D|) + |D|^2 \cdot |S| + |D|) \quad (24)$$

In practice, the constant factors for sorting are much smaller than for the deviation calculation. For high-dimensional data sets, we can also assume that $|S| \ll |D|$. This is because GMD searches for lower-dimensional projections of D. Hence, the runtime of GMD mainly depends on the number of subspaces the deviation is calculated for. It is quadratic in the number of dimensions.

5.3.2 Complexity of outlier detection in subspaces

The number of subspaces output is at most $|D|$. Consequently, the subsequent outlier detection algorithm is applied to $|D|$ subspaces. The complexity of outlier detection algorithms usually depends on N. Consequently, the runtime complexity of the subspace search is small compared to the outlier detection because $D \ll N$ for most applications. Our experiments will confirm these theoretical considerations.

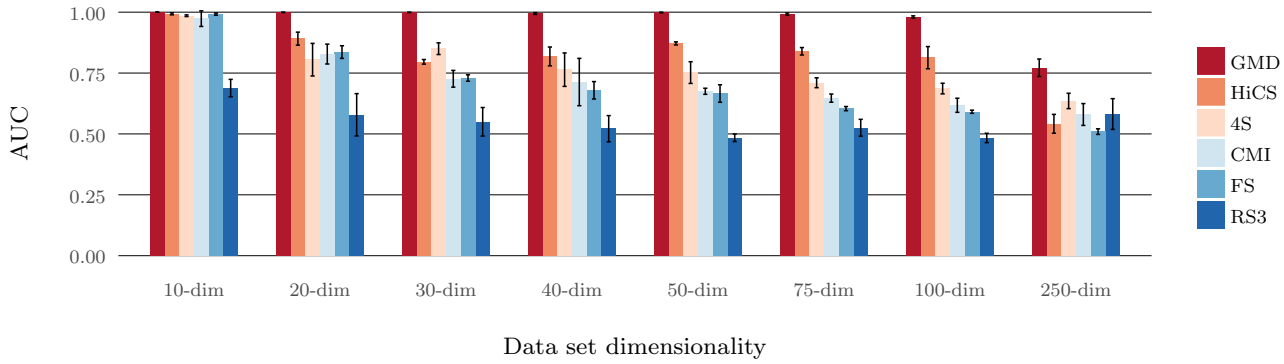

Fig. 2 Experimental results for 24 different synthetic data sets. The figure reports the average AUC and standard deviation grouped by the data set dimensionality

6 Experiments

We evaluate the dimension-based approach with experiments on synthetic and real world data sets. Our focus is to compare the dimension-based framework with purely correlation-based approaches.

6.1 Experiment setup

We benchmark GMD against HiCS [7], CMI [19] and 4S [18]. For comparability to earlier benchmark results [7], we use LOF [3] as the outlier detection model. In addition, we also run LOF on the Full Space (FS) and on randomly selected three-dimensional subspaces (RS3) as a baseline. For RS3, we set the number of subspaces to the data set dimensionality. We provide data sets, preprocessing scripts, our reference implementation and our experiment framework to facilitate the reproducibility of our experiments[1]. For CMI and 4S, we use the respective implementations provided by the authors.

The deviation function used by GMD and by HiCS relies on two parameters. α sets the expected number of data objects in a subspace slice after conditioning on C_{-i}. M is the number of Monte Carlo iterations. We use the settings $\alpha = 0.1$ and $M = 100$ for both GMD and HiCS.

The HiCS framework as well as CMI make use of two search beams to identify the subspace candidates. The first beam stores the subspaces with highest contrast for each Apriori level and is used to generate the candidates for the next iteration. Its size is the *candidate_cutoff* parameter. Increasing *candidate_cutoff* can expand the coverage of the search space. However, choosing a good cutoff for high dimensional data is not intuitive. The second beam limits the number of subspaces in the final result which we refer to as *output_cutoff*. In the synthetic data benchmark, we follow the author recommendations and set *candidate_cutoff* for HiCS to 500 and for CMI to 400. The number of output subspaces is set to the top-100 subspaces with the highest contrast in both cases. For real world data, we increase *candidate_cutoff* to 5000 and *output_cutoff* to 1000 to better cope with the high number of dimensions. For 4S, we use the version with Multi-Pruning. For all approaches, we remove duplicate subspaces in the result.

In each run of LOF, we vary the $MinPts$ parameter from 10 to 100 with step size 10 and take the result with the highest AUC. This reduces the dependence of the result on the LOF parameter setting.

6.2 Evaluation metrics

We use different metrics to evaluate a subspace-search method.

6.2.1 Search-result quality

We use the widely used Area Under the ROC Curve (AUC) to assess the outlier ranking. This metric describes how well outliers are ranked relatively to inliers.

6.2.2 Robustness

A subspace search is robust if it is insensitive to (a) the properties of the data set and (b) to the search parameter settings. For synthetic data sets, to study (a), there are three different versions created for each dimensionality independently. For real world data, we create several versions of an initial data set by sampling down to different outlier percentages multiple times. Each of the data sets created is used as a fair comparison between subspace search approaches. We deem an approach more robust if it outperforms its competitor on most of the data sets.

[1] `https://www.ipd.kit.edu/trittenb/gmd/readme`

Table 3 Average number of Subspaces returned by the subspace search for the synthetic data sets

Dataset	GT	GMD	HiCS	4S	CMI	FS	RS3
10-dim	3	5	100	6	54	1	10
20-dim	6	14	100	12	100	1	20
30-dim	9	22	100	20	100	1	29
40-dim	11	31	100	29	100	1	40
50-dim	14	40	100	38	100	1	49
75-dim	23	57	100	60	100	1	75
100-dim	29	78	100	80	100	1	100
250-dim	115	139	100	111	100	1	250

6.2.3 Number of subspaces

Recent research [2,15] has observed that outlier-detection methods should not only return the outliers themselves, but also additional information allowing to comprehend these results. With object-based methods in particular [11,16,26], this additional information for an object is a subspace where it is outlying. If the number of subspaces explaining the outlierness of all objects in the result is small, then the result as a whole is easier to comprehend, i.e., has higher interpretability. In addition, a small number of subspaces returned is good because of the multiplicative impact on the runtime. This is because outlier detection runs on each subspace.

6.2.4 Runtime

Subspace search and outlier detection are decoupled processing steps. We therefore measure the runtimes separately. We limit the runtime comparison to our R/C++ implementation of GMD and HiCS. This makes both algorithms directly comparable. Runtime comparison with different programming languages and frameworks used in [18,19] is not meaningful.

6.3 Experiments on synthetic data

We use synthetic data sets which several outlier related benchmarks [7,14,5,25] have used. The data has dimensionalities from 10 to 100. For each dimensionality there are three randomly generated data sets. In each of them, randomly selected subspaces of dimensionality between 2 and 5 are artificially correlated. In each of these subspaces, 5 objects have been placed that deviate from the general distribution in that subspace. These objects are placed in a way that they are not visible in any of the lower-dimensional projections of that subspace. Each outlier can also appear in several subspaces.

We also use three randomly generated data sets of 250 dimensions. There, 50 dimensions contain 2-5 dimensional correlated subspaces with 6 outliers. We have placed them in each subspace in the same manner as with the other synthetic data sets. Among the remaining 200 dimensions, two-dimensional subspaces of high correlation have been created without any additional outliers.

Search-result quality and robustness: Figure 2 graphs the average AUC and its standard deviation. For all versions of the data sets, the dimension-based framework outperforms the competitors. Hence, GMD is the more robust approach. For data sets between 10 and 100 dimensions, GMD identifies the relevant subspaces almost perfectly. As expected, the full space approach is only slightly above a random guess with increased dimensionality, as outliers are hidden in the full space. For the 250-dimensional data sets, HiCS fails to identify relevant subspaces. A reason might be that highly correlated two-dimensional projections take over the search beam and prevent HiCS from identifying the important subspaces that contain outliers. GMD instead identifies a large share of the relevant subspaces.

Number of subspaces: Table 3 lists the number of subspaces identified by the search approaches. The ground truth (GT) is the number of subspaces that are known to be artificially correlated.

GMD achieves higher AUC with significantly fewer subspaces than HiCS. It might be possible to improve HiCS by optimizing its cutoff parameters. In general, however, we expect the AUC to decline with a reduced search beam size due to inflated search results (cf. Example 3). For example, setting $output\_cutoff = 20$ for the 30-dim data set, which is the number of subspaces GMD returns, worsens the AUC of HiCS. CMI also makes use of cutoff parameters, and the interpretation of the results is similar to the one of HiCS. The number of subspaces for 4S is comparable to GMD. However, results are significantly worse.

Runtime: Average runtimes are less than 300 seconds for most of the data sets. This is low compared to the real world experiments. The runtime comparison on

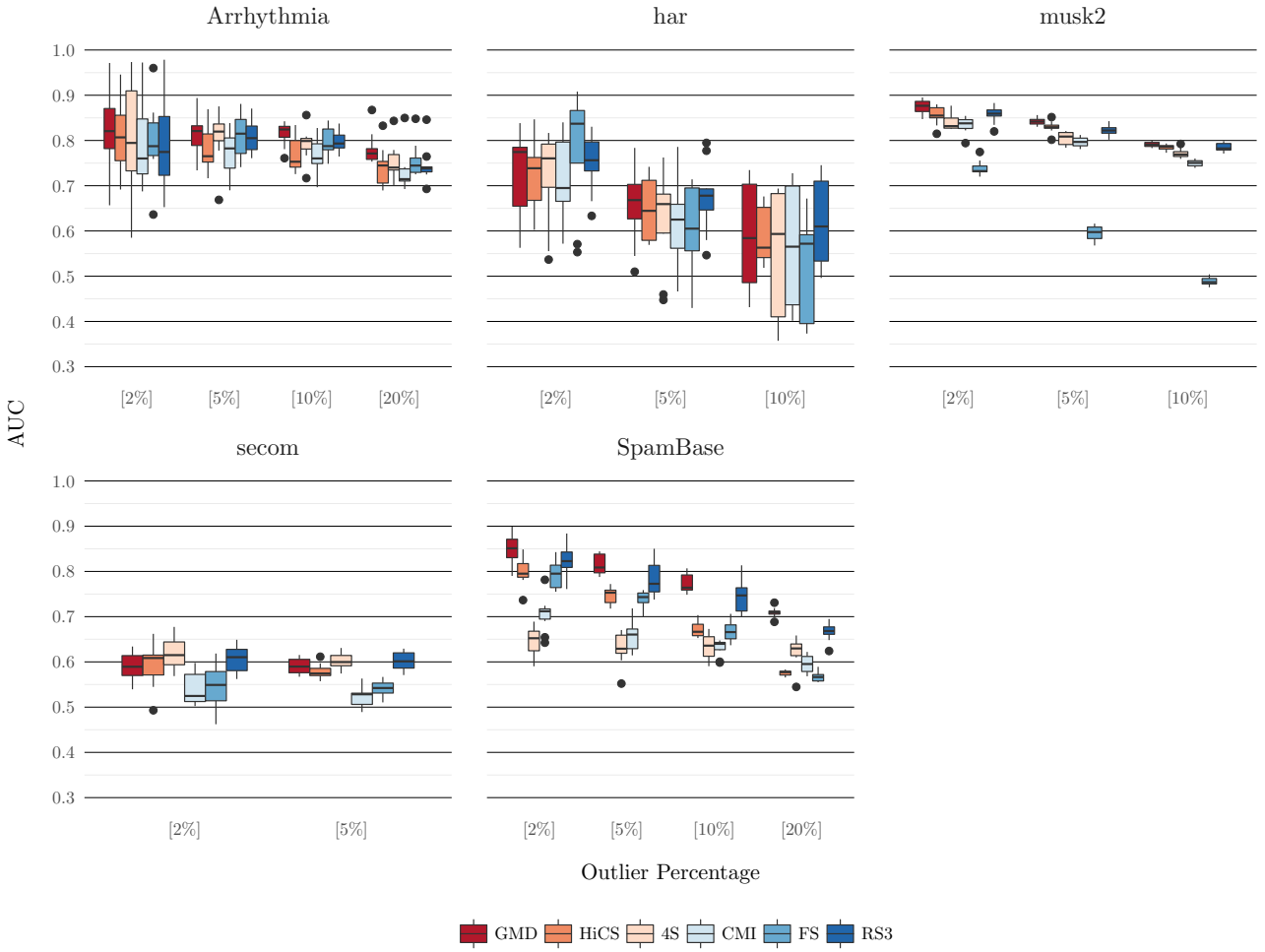

Fig. 3 Experimental results on high dimensional real world data sets. Each data set is resampled multiple times to different percentages of outliers to produce a meaningful benchmark

synthetic data sets is not overly insightful because of a small number of observations and the small search beam size. We provide a more extensive runtime evaluation for real world data in Section 6.4.

6.4 Experiments on real world data

Benchmarking outlier-detection methods on real world data is challenging, as there are only few data sets publicly available. Most of these data sets are adapted from related disciplines, such as classification, by downsampling one class to represent the rare, outlying objects. This preprocessing introduces a bias. The resulting data set depends on which class is selected as the outlying class and on whether the downsampling actually produces outliers or just decreases object densities.

Search-result Quality: Recently, Campos et al. have pusblished a study on outlier-detection benchmarking [4]. They propose to do the downsampling in different ways, e.g., regarding different outlier percentages. In this way, the benchmark runs on multiple versions of the same data set, to make the preprocessing bias visible. From the benchmark repository used in [4], only the data sets Arrhythmia and SpamBase have a sufficient number of dimensions to be considered high-dimensional. So we use further data sets. These are the Human Activity Recognition (har) data, the SECOM data set and the musk (version 2) data set from the UCI Machine Learning Repository [13]. We preprocess them similarly to [4]. Table 4 features the number of observations before preprocessing and the dimensionalities of the data sets. Table 5 lists the number of variants after the downsampling. We run our experiments on the normalized versions of the data sets to be able to also compare our approach to 4S and CMI.

Due to the large number of individual experiment settings described so far, we sample data sets with $N >$

Table 4 Average number of Subspaces returned by the subspace search for the real world data sets

Dataset	$\|D\|$	N	GMD	HiCS	4S	CMI	FS	RS3
Arrhythmia	260	452	198-214	1000	42-57	1000	1	257-259
har	561	10299	471-493	1000	19-27	1000	1	559-561
musk2	166	6598	146-157	1000	11-14	1000	1	165-166
secom	538	1567	300-307	1000	67-75	1000	1	421-422
SpamBase	58	4601	39-47	1000	12-30	1000	1	54-57

Table 5 Listing of the percentage how often the algorithm performs best on a variant of a dataset. The second column (#Var) is the number of resampling variants of the dataset

Dataset	#Var	GMD	HiCS	4S	CMI	FS	RS3
Arrhythmia	40	0.45	0.10	0.15	0.02	0.10	0.18
har	30	0.00	0.30	0.00	0.00	0.33	0.37
musk2	30	0.93	0.00	0.03	0.00	0.00	0.03
secom	20	0.10	0.10	0.55	0.00	0.05	0.20
SpamBase	40	0.92	0.00	0.00	0.00	0.00	0.08

1000 down to 1000 observations to reduce runtime for the subspace-search approaches.

Figure 3 graphs the resulting AUC over the different versions of the same data set. The results are visualized as boxplots for different percentages of outliers. GMD yields the best performance on the Arrhythmia, musk2 and SpamBase data sets. On the har data set, none of the subspace-search approaches is significantly better than FS or RS3. For the secom data set, the correlation-based 4S results in a better search-result quality than GMD. However, all approaches are close to an AUC of 0.5, i.e., results are not much better than a random outlier ranking.

Robustness: Besides the distribution of AUC values over all data sets, it is important to also study the direct comparison on each version of the data set separately. Table 5 compares the various algorithms. GMD is the most robust one on Arrhythmia, musk2 and SpamBase. 4S is the most robust approach on the secom data set. For har, the most robust approach is the baseline RS3 followed by the baseline FS.

Number of subspaces: We now look at the number of subspaces identified with each approach. For HiCS and CMI to deal with high-dimensional data, the cut-off parameters must be large enough. However, finding a good values for these is not intuitive, and the choice can only be evaluated ex-post. Due to the long runtimes, we have refrained from trying variations exhaustively. In our experiments, the number of spaces output is smaller for GMD than for HiCS and CMI. 4S yields the smallest number of subspaces. However, the small number of subspaces comes with a low search-result quality for many of the data sets. We hypothesize that this is because 4S might miss relevant subspaces and only detects a few outliers correctly. In this case, the low search-result quality foils the benefit of a small number of subspaces.

Runtime: For the real world data sets, we also measure the runtime of the subspace search and of the actual outlier detection. Figure 4 depicts the average runtimes. GMD results in a lower runtime overall for all data sets when comparing to HiCS. The reason is the number of subspaces that need to be searched for outliers and the complexity of the outlier-detection algorithm. For LOF, the runtime is in $O(N^2)$, i.e., it increases with the number of observations. Because the outlier-detection algorithm needs to run on every subspace, their number affects the overall runtime significantly. The time to search for subspaces in turn is almost negligible for an increasing number of observations. Other approaches that produce only few subspaces, such as FS and 4S, can also result in low runtimes for subsequent outlier detection. This is particularly true since subspace search is effective and efficient on samples of the data.

7 Conclusions

Purely correlation-based subspace search aims to find all subspaces of a data set where outliers are likely to occur. These approaches use the correlation of the subspace dimensions to quantify the potential that a subspace contains outliers. A major weakness of these approaches is that correlation is independent of the specific subspace dimensions. As a consequence, some of the data set dimensions might not be represented in the search result.

In this paper, we have proposed a dimension-based evaluation of subspaces, allowing for a more differentiated view on a subspace. Using this evaluation, we

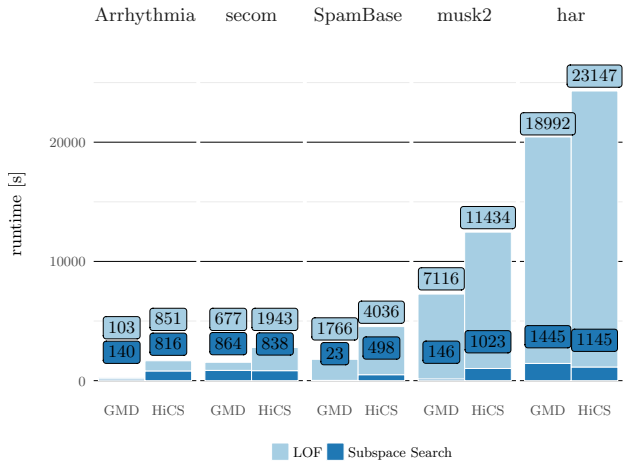

Fig. 4 Average runtime on real world data

have introduced a notion of dominance, to make sets of subspaces directly comparable. We have proposed GMD, a heuristic that builds upon the dimension-based framework. It is a greedy algorithm that identifies the relevant set of subspaces. Comprehensive experiments on synthetic and real world data sets demonstrate the advantages of our dimension-based approach for high-dimensional data sets.

Acknowledgements This work was supported by the German Research Foundation (DFG) as part of the Research Training Group GRK 2153: *Energy Status Data – Informatics Methods for its Collection, Analysis and Exploitation.*

Conflict of interest On behalf of all authors, the corresponding author states that there is no conflict of interest.

References

1. Aggarwal, C., Sathe, S.: Theoretical foundations and algorithms for outlier ensembles. SIGKDD Explor. Newsl. **17**(1), 24–47 (2015)
2. Angiulli, F., Fassetti, F., Manco, G., Palopoli, L.: Outlying property detection with numerical attributes. Data Min. Knowl. Discov. pp. 134–163 (2016)
3. Breunig, M.M., Kriegel, H.P., Ng, R.T., Sander, J.: LOF: Identifying density-based local outliers. SIGMOD **29**(2), 93–104 (2000)
4. Campos, G.O., Zimek, A., Sander, J., Campello, R.J.G.B., Micenková, B., Schubert, E., Assent, I., Houle, M.E.: On the evaluation of unsupervised outlier detection: measures, datasets, and an empirical study. Data Min. Knowl. Discov. pp. 1–37 (2016)
5. Duan, L., Tang, G., Pei, J., Bailey, J., Campbell, A., Tang, C.: Mining outlying aspects on numeric data. Data Min. Knowl. Discov. **29**(5), 1116–1151 (2015)
6. Duan, L., Tang, G., Pei, J., Bailey, J., Dong, G., Nguyen, V., Campbell, A., Tang, C.: Efficient discovery of contrast subspaces for object explanation and characterization. Knowl. Inf. Syst. **47**(1), 99–129 (2015)
7. Keller, F., Müller, E., Böhm, K.: HiCS: High contrast subspaces for Density-Based outlier ranking. In: ICDE, pp. 1037–1048 (2012)
8. Keller, F., Müller, E., Wixler, A., Böhm, K.: Flexible and adaptive subspace search for outlier analysis. In: CIKM, pp. 1381–1390 (2013)
9. Knorr, E.M., Ng, R.T.: Finding intensional knowledge of distance-based outliers. In: VLDB, vol. 99, pp. 211–222 (1999)
10. Kriegel, H.P., Kröger, P., Schubert, E., Zimek, A.: Outlier detection in Axis-Parallel subspaces of high dimensional data. In: PAKDD (2009)
11. Kriegel, H.P., Kroger, P., Schubert, E., Zimek, A.: Outlier detection in arbitrarily oriented subspaces. In: ICDM, pp. 379–388 (2012)
12. Kriegel, H.P., Schubert, M., Zimek, A.: Angle-based outlier detection in high-dimensional data. In: KDD, pp. 444–452 (2008)
13. Lichman, M.: UCI machine learning repository (2013). URL `http://archive.ics.uci.edu/ml`
14. Micenková, B., Dang, X.H., Assent, I., Ng, R.T.: Explaining outliers by subspace separability. In: ICDM, pp. 518–527 (2013)
15. Müller, E., Keller, F., Blanc, S., Böhm, K.: OutRules: A framework for outlier descriptions in multiple context spaces. In: ECML PKDD, pp. 828–832 (2012)
16. Müller, E., Schiffer, M., Seidl, T.: Statistical selection of relevant subspace projections for outlier ranking. In: ICDE, pp. 434–445 (2011)
17. Nguyen, H.V., Ang, H.H., Gopalkrishnan, V.: Mining outliers with ensemble of heterogeneous detectors on random subspaces. DASFAA pp. 368–383 (2010)
18. Nguyen, H.V., Müller, E., Böhm, K.: 4S: Scalable subspace search scheme overcoming traditional apriori processing. In: Big Data Conference, pp. 359–367 (2013)
19. Nguyen, H.V., Müller, E., Vreeken, J., Keller, F., Böhm, K.: CMI: An Information-Theoretic contrast measure for enhancing subspace cluster and outlier detection. In: ICDM, pp. 198–206 (2013)
20. Pang, G., Cao, L., Chen, L., Liu, H.: Unsupervised feature selection for outlier detection by modelling hierarchical Value-Feature couplings. In: ICDM (2016)
21. Pang, G., Cao, L., Chen, L., Liu, H.: Learning homophily couplings from Non-IID data for joint feature selection and Noise-Resilient outlier detection. In: IJCAI (2017)
22. Pasillas-Díaz, J.R., Ratté, S.: Bagged subspaces for unsupervised outlier detection. Comput. Intell. (2016)
23. Pestov, V.: On the geometry of similarity search: Dimensionality curse and concentration of measure. Inf. Process. Lett. **73**(1), 47–51 (2000)
24. Sathe, S., Aggarwal, C.C.: Subspace outlier detection in linear time with randomized hashing. In: ICDM, pp. 459–468 (2016)
25. Vinh, N.X., Chan, J., Romano, S., Bailey, J., Leckie, C., Ramamohanarao, K., Pei, J.: Discovering outlying aspects in large datasets. Data Min. Knowl. Discov. pp. 1–36 (2016)
26. Zhang, J., Gao, Q., Wang, H.: A novel method for detecting outlying subspaces in high-dimensional databases using genetic algorithm. In: ICDM, pp. 731–740 (2006)
27. Zimek, A., Campello, R.J.G.B., Sander, J.: Ensembles for unsupervised outlier detection: Challenges and research questions a position paper. SIGKDD Explor. Newsl. **15**(1), 11–22 (2014)
28. Zimek, A., Schubert, E., Kriegel, H.P.: A survey on unsupervised outlier detection in high-dimensional numerical data. Stat. Anal. Data Min. **5**(5), 363–387 (2012)