Dimension reduction for outlier detection using DOBIN

Sevvandi Kandanaarachchi, Rob J. Hyndman

August 20, 2019

Abstract

This paper introduces DOBIN, a new approach to select a set of basis vectors tailored for outlier detection. DOBIN has a solid mathematical foundation and can be used as a dimension reduction tool for outlier detection tasks. We demonstrate the effectiveness of DOBIN on an extensive data repository, by comparing the performance of outlier detection methods using DOBIN and other bases. We further illustrate the utility of DOBIN as an outlier visualization tool. The R package *dobin* implements this basis construction.

1 Introduction

Outlier detection is used in diverse applications such as the identification of extreme weather events, stock market crashes and fraudulent transactions.

A common challenge in many domains is that a data point may be an outlier in the original high dimensional space, but not an outlier in the low dimensional subspace created by a dimension reduction method. A growing body of literature on subspace outlier detection focuses on meeting this challenge (e.g., Aggarwal & Yu 2001, Keller et al. 2012).

We address this challenge by proposing a set of basis vectors tailored for unsupervised outlier detection which we call DOBIN: a Distance based Outlier BasIs using Neighbours. The acronym DOBIN is inspired from the informal verb 'dob in', which the Collins English dictionary defines as 'to inform against or report, specially to the police'. We emphasize that DOBIN is not an outlier detection method; rather it is a pre-processing step that can be used by any outlier detection method.

To the best of our knowledge no formal dimension reduction techniques exist for outlier detection. It is common to use Principal Component Analysis (PCA) in instances where dimension reduction is used when detecting outliers (e.g., Talagala et al. 2019, Hyndman et al. 2015). However, the goal of PCA is to find a set of basis vectors that explains the variance of the data, such that the highest variance is in the direction of the first vector, and so on. Therefore, the PC basis may not be suited for detecting outliers. By introducing DOBIN, we aim to bridge this gap in the literature.

We envisage two uses of DOBIN. First, as a basis more conducive to outlier detection, it brings outliers to the forefront using a smaller number of components. In this sense, it is somewhat similar to subspace outlier detection, even though DOBIN does not detect outliers directly. If outliers arise in the full input space, DOBIN enables the outlier detection algorithm to detect outliers using fewer components. The second use of DOBIN is to assist with visualisation of outliers as we will see in Section 4.

The literature on outlier detection has evolved in two fields, with contributions from statisticians and computer scientists. Although the methodologies used by the two groups are somewhat different, both groups have contributed to a rich and diverse literature, benefiting the broader community. Useful summaries of the computer science literature are provided by Goldstein & Uchida (2016) and Zimek et al. (2012); while similar surveys of the statistics outlier literature are given in Rousseeuw & Leroy (2005) and Unwin (2019*a*). We have included methods from both disciplines in this paper.

Some methods treat outlier identification as a binary rule, with observations classified as outliers or otherwise (e.g., Billor et al. 2000, Wilkinson 2017, Rousseeuw & Bossche 2018), while other methods rank all observations in terms of "outlyingness" (e.g., Breunig et al. 2000, Liu et al. 2008), or provide a probability of an observation being an outlier (e.g., Kriegel et al. 2009). This difference impacts the performance evaluation metrics. Metrics such as false negatives and false positives or positive predictive value and negative predictive value are used to evaluate the binary

results (Wilkinson 2017). By contrast, having a ranking of the observations leads to use tools such as area under the Receiver Operator Characteristic (ROC) curve, area under the Precision-Recall (PR) curve or precision at n to evaluate performance (Campos et al. 2016).

In Sections 3 and 5 we investigate the effect of DOBIN on three outlier detection methods which first appeared in the computer science literature: LOF (Breunig et al. 2000), KNN (Ramaswamy et al. 2000) and iForest (**?**). To evaluate their performance, we use area under the ROC curve. We examine the effectiveness of DOBIN by comparing it against two other bases: PCA and the original basis. In Section 3 we use synthetic data to conduct experiments and in Section 5 we use real data from our repository of more than 12,000 datasets (Kandanaarachchi et al. 2019).

Section 4 examines outlier detection methods developed by the statistical community on some associated datasets. We use O3 plots by Unwin (2019*a*), which can be used to compare the results of six different outlier detection methods: *HDoutliers* (Wilkinson 2017), *mvBACON* (Billor et al. 2000), *adjOutlyingness* (Brys et al. 2005), *covMcd* (Rousseeuw & Driessen 1999) , *FastPCs* (Vakili & Schmitt 2014) and *DetectDeviatingCells* (Rousseeuw & Bossche 2018). We use O3 plots in conjunction with the first two DOBIN components of each dataset and visually inspect if the outliers identified by the O3 plots are further away from the other points in the DOBIN space. This section highlights the usage of DOBIN as an outlier visualization tool.

We have produced an R package *dobin* (Kandanaarachchi 2019), which computes the DOBIN basis. In addition, all examples in this paper are available in the supplementary materials at `https://github.com/sevvandi/Outlier-Basis`. We start the next section with the mathematical framework of DOBIN.

2 Mathematical Framework

Let $X_{N\times p}$ be a matrix denoting a dataset of N observations and p attributes. Let us denote the i^{th} row of X by $\boldsymbol{x}_i$. The distance between the points $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ can be written as

$$\text{dist}(\boldsymbol{x}_i, \boldsymbol{x}_j)^2 = \left(\boldsymbol{x}_i - \boldsymbol{x}_j\right)^\top S \left(\boldsymbol{x}_i - \boldsymbol{x}_j\right) , \tag{1}$$

where S is a symmetric positive definite matrix. In addition when S is diagonal, i,e, $S = \text{diag}(s_1, s_2, \ldots, s_p)$ we obtain

$$\text{dist}(\boldsymbol{x}_i, \boldsymbol{x}_j)^2 = \left\langle \eta, \left(\boldsymbol{x}_i - \boldsymbol{x}_j\right) \circ \left(\boldsymbol{x}_i - \boldsymbol{x}_j\right) \right\rangle , \tag{2}$$

where $\langle \cdot, \cdot \rangle$ denotes the standard inner product in $\mathbb{R}^p$, $\eta = \left(s_1, s_2, \ldots, s_p\right)^\top$, and $\circ$ denotes the element-wise vector product. As S is symmetric and positive definite it is diagonalizable, thus giving us motivation for considering a diagonal S.

Next we establish the Y space, which is instrumental in the construction of DOBIN.

2.1 The Y space

Let $\boldsymbol{y}_{ij} = \left(\boldsymbol{x}_i - \boldsymbol{x}_j\right) \circ \left(\boldsymbol{x}_i - \boldsymbol{x}_j\right) = \left(\left(x_{i1} - x_{j1}\right)^2, \left(x_{i2} - x_{j2}\right)^2, \ldots, \left(x_{ip} - x_{jp}\right)^2\right)$. Substituting in equation (2) gives

$$\text{dist}(\boldsymbol{x}_i, \boldsymbol{x}_j)^2 = \left\langle \eta, \boldsymbol{y}_{ij} \right\rangle . \tag{3}$$

By finding the appropriate η of unit length, we can maximize the distance between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$. This prompts the question as to which $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ need to be chosen to maximize the distance between them, if outlier detection is the goal. If we use knn distances as a guiding principle, we are interested in the k nearest neighbours of a given point. Consequently, we can choose $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ that share a common neighbourhood. Also we note that if we were to calculate $\boldsymbol{y}_{ij}$ for all pairs, it would be computationally expensive as there are $N(N+1)/2$ pairs to consider. In addition, this would give us pairwise distances between points that we are not interested in, such as points on the boundary of the point cloud, that are opposite each other. Finding η that maximizes the distances between such points is not beneficial for outlier detection, because a large distance between polar-opposite points does not mean that either of those points are outliers. Therefore we do not compute $\boldsymbol{y}_{ij}$ for all pairs of $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$; rather we select the pairs that we want to include in our Y space.

For each point $\boldsymbol{x}_i$ we compute $\boldsymbol{y}_{ij}$ arising from a set a nearest neighbours. From Kandanaarachchi et al. (2018) we know that nearest neighbours depend on the normalization technique. As a result, we choose two normalization

techniques to pre-process the data; Min-Max and Median-IQR. Min-Max normalization scales each column to values between 0 and 1, with the minimum mapped to 0 and the maximum mapped to 1. Median-IQR scales each column to median 0 and IQR 1. Let X_1 denote the normalized data using Min-Max and X_2 using Median-IQR. For each normalization technique we choose a set of nearest neighbours from k_1 to k_2, that is for any point $\boldsymbol{x}_i$ we choose the k_1^{th} nearest neighbour to the k_2^{th} nearest neighbour. Here k_1 and k_2 are parameters and one can choose $k_1 = 1$.

Next, for each point we find the union of nearest neighbours computed using the two normalization techniques. For example, for a given point $\boldsymbol{x}_i$, if the nearest neighbours according to Min-Max are $\{\boldsymbol{x}_2, \boldsymbol{x}_5, \boldsymbol{x}_9\}$ and Median-IQR gives $\{\boldsymbol{x}_5, \boldsymbol{x}_4, \boldsymbol{x}_7\}$, then we consider the union of these two sets as the set of nearest neighbours of $\boldsymbol{x}_i$. Using the nearest neighbours we construct a list of pairs denoted by T. Continuing the same example, the pairs $(\boldsymbol{x}_i, \boldsymbol{x}_2)$, $(\boldsymbol{x}_i, \boldsymbol{x}_5)$, $(\boldsymbol{x}_i, \boldsymbol{x}_4)$, $(\boldsymbol{x}_i, \boldsymbol{x}_7)$, $(\boldsymbol{x}_i, \boldsymbol{x}_9)$, and so on will be in T.

For each pair in T, we compute $\boldsymbol{y}_{ij}$ as in equation (2) on the normalized space X_1 or X_2, depending on user preference. We now have the initial Y space: a set of $\tilde{M}$ points in $\mathbb{R}^{p^+}$. As the list of pairs that give rise to the initial Y space are contained in T, we can denote $Y = \{\boldsymbol{y}_\ell\}_{\ell=1}^{\tilde{M}}$, where each $\boldsymbol{y}_\ell = \boldsymbol{y}_{i_{(\ell)}j_{(\ell)}}$, i.e. ℓ is the row number in the matrix Y and each $\boldsymbol{y}_\ell$ comes from the ℓ^{th} pair in T.

We are interested in neighbouring distances that are relatively large. The Euclidean distance squared between two points in the X_1 or X_2 space can be written as $\sum_{m=1}^{p} y_{\ell m}$ for the appropriate ℓ. In order to consider only points with relatively large distances, we remove points $\boldsymbol{y}_\ell$ that contribute to distances below a certain threshold determined by a percentile, i.e. we remove points $\boldsymbol{y}_\ell$ such that $\sum_{m=1}^{p} y_{\ell m} < Q$ where Q is the q^{th} percentile of $\left\{\sum_{m=1}^{p} y_{\ell m}\right\}_{\ell=1}^{\tilde{M}}$. We remove the associated pairs from T as well. This constitutes the Y space: $Y = \{\boldsymbol{y}_\ell\}_{\ell=1}^{M}$.

The main advantage of the Y space is that it makes distance computation between two points in the X space a linear function. This makes maximizing distances between points a much easier task. Algorithm 1 summarizes the construction of the Y space.

input : X, $k_1, k_2 \in \mathbb{Z}^+$, $q \in (0,1)$ and choice of normalization.
output : The space Y consisting of all $\{\boldsymbol{y}_\ell\}_{\ell=1}^{M}$ and the associated indices i and j.

1 Normalize X using Min-Max normalization. Let us call this space X_1.
2 Find k_1 to k_2 nearest neighbours for each point in X_1.
3 Next normalize X using Median-IQR normalization. Let us call this space X_2.
4 Find k_1 to k_2 nearest neighbours for each point in X_2.
5 Then for each point collate all unique neighbours from both Min-Max and Median-IQR normalization methods.
6 Let T = the set of i and j indices, corresponding to the pairs of points in X.
7 For all these pairs, compute $\boldsymbol{y}_{ij}$ as in equation (3) using X_1 or X_2 space according to the choice of normalization. Let $Y = \left\{\boldsymbol{y}_1 = \boldsymbol{y}_{i_{(1)}j_{(1)}}, \boldsymbol{y}_2 = \boldsymbol{y}_{i_{(2)}j_{(2)}}, \ldots, \boldsymbol{y}_{\tilde{M}} = \boldsymbol{y}_{i_{(\tilde{M})}j_{(\tilde{M})}}\right\}$, where $\left(i_{(1)}, j_{(1)}\right)$ is the first $(i,j)^{\text{th}}$ pair in T. Consequently Y is an $\tilde{M} \times p$ matrix, with the ℓ^{th} row denoted by $\boldsymbol{y}_\ell$.
8 Let the q^{th} percentile of $\left\{\sum_{m=1}^{p} y_{\ell m}\right\}$ be Q. The quantity $\sum_{m=1}^{p} y_{\ell m}$ is the distance between points $\boldsymbol{x}_{i_{(\ell)}}$ and $\boldsymbol{x}_{j_{(\ell)}}$ in X_1 or X_2 space.
9 Remove points $\boldsymbol{y}_\ell$ for which $\sum_{m=1}^{p} y_{\ell m} < Q$.
10 Remove the associated pairs of (i,j) from T.
11 The remaining points $\boldsymbol{y}_\ell$ constitute the Y space: $Y = \{\boldsymbol{y}_\ell\}_{\ell=1}^{M}$.

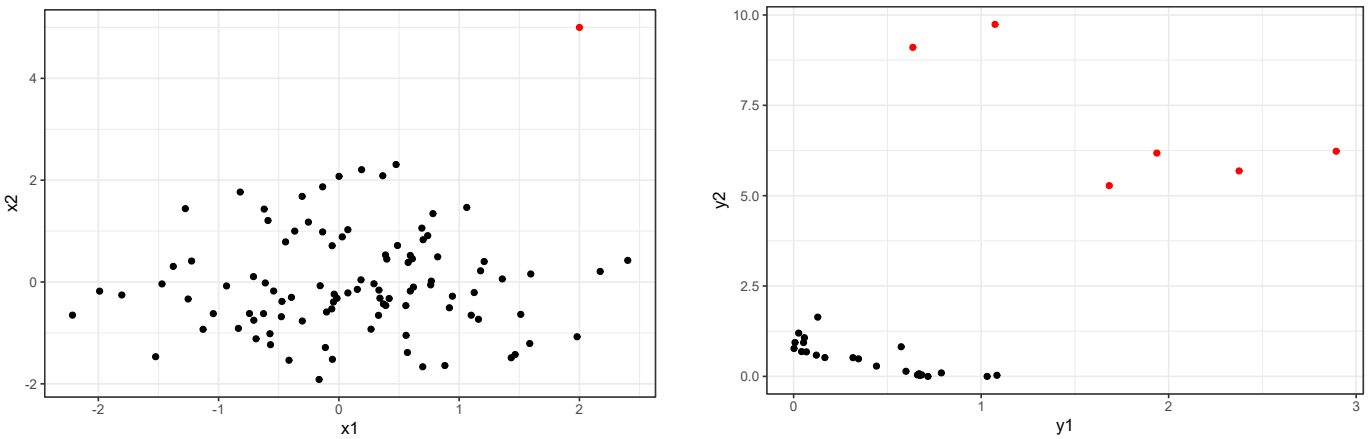
Algorithm 1: *Construction of the Y space.*

Figure 1 illustrates the X and Y spaces where X is a 2 dimensional normal distribution of 100 points with a single outlier depicted in red. We see that the Y space consists of a lesser number of points compared to X, of which 6 points are contributed by the outlier. This is because k_1 to k_2 neighbours are considered in the construction of the Y space and the points $\boldsymbol{y}_k$ with relatively small distances ($\sum_i y_{ki} \leq Q$) are removed. Thus, the effect of outliers is magnified in the Y space.

What i k_1 and k_2 in this ex ample'

We summarise the salient features of the Y space below:

1. $Y = \{\boldsymbol{y}_\ell\}_{\ell=1}^{M}$, where $\boldsymbol{y}_\ell \in \mathbb{R}^{p^+}$.
2. Each $\boldsymbol{y}_\ell \in Y$ is constructed from two points $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ in X with $T(\ell) = (i,j)$, where T is a list consisting of the pairs in X that are used to construct Y. In addition, $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are k-nearest neighbours for some $k \in \{k_1, \ldots, k_2\}$.
3. For i, j, l as in above, $\boldsymbol{y}_\ell = \left(\boldsymbol{x}_i - \boldsymbol{x}_j\right) \circ \left(\boldsymbol{x}_i - \boldsymbol{x}_j\right)$, where $\circ$ denotes the element-wise vector product.
4. The Y space contains points $\boldsymbol{y}_\ell$ with relatively high $\sum_{m=1}^{p} y_{\ell m}$, i.e. it contains vectors that contribute to high knn distances in the X space.



(a) (b)

Figure 1: *The X and Y spaces. The X space in (a) and the associated Y space in (b) with* $q = 0.95$*. The red coloured point in the X space gives rise to 6 points, which are coloured in red in the Y space.*

2.2 Maximising the distance between points

Restating equation (3) using $T(\ell) = (i, j)$ we obtain

$$\sum_{(i,j)\in T} \text{dist}(\boldsymbol{x}_i, \boldsymbol{x}_j)^2 = \sum_{\ell} \langle \eta, \boldsymbol{y}_\ell \rangle \ . \tag{4}$$

Thus the total distance squared for relatively high nearest neighbour distances $\sum_{(i,j)\in T} \text{dist}(\boldsymbol{x}_i, \boldsymbol{x}_j)^2$ can be maximised by finding the appropriate η which maximises $\sum_\ell \langle \eta, \boldsymbol{y}_\ell \rangle$. However, by choosing $\eta_1 = c\eta_0$, with $c > 1$ will automatically increase the value of $\sum_\ell \langle \eta, \boldsymbol{y}_\ell \rangle$ by c. Thus, we restrict η such that $\|\eta\| = 1$. We now state our optimisation problem as

$$\begin{aligned} &\max_{\eta} \sum_{\ell} \langle \eta, \boldsymbol{y}_\ell \rangle \ , \\ &\text{subject to} \quad \|\eta\|^2 = 1 \, . \end{aligned} \tag{5}$$

Solving this using Lagrange Multipliers we obtain $\nabla f = \lambda \nabla g$, with

$$\begin{aligned} f(\eta) &= \sum_{\ell} \langle \eta, \boldsymbol{y}_\ell \rangle \ , \text{and} \\ g(\eta) &= \|\eta\|^2 - 1 = 0 \, . \end{aligned} \tag{6}$$

As $\nabla f = \left(\frac{\partial f}{\partial \eta_1}, \frac{\partial f}{\partial \eta_2}, \cdots, \frac{\partial f}{\partial \eta_p} \right)$ we compute

$$\begin{aligned} \frac{\partial f}{\partial \eta_j} &= \frac{\partial}{\partial \eta_j} \sum_{\ell} \langle \eta, \boldsymbol{y}_\ell \rangle \ , \\ &= \frac{\partial}{\partial \eta_j} \sum_{\ell=1}^{M} \sum_{i=1}^{p} \eta_i y \, , \\ &= \sum_{\ell=1}^{M} \sum_{i=1}^{p} \delta_{ij} y \, , \\ &= \sum_{\ell=1}^{M} y \, . \end{aligned} \tag{7}$$

where δ_{ij} is the Kronecker delta function which equals 1 when $i = j$ and zero otherwise. Hence

$$\nabla f = \left(\sum_{\ell=1}^{M} y_{\ell 1}, \sum_{\ell=1}^{M} y_{\ell 2}, \ldots, \sum_{\ell=1}^{M} y \right) = \sum_{\ell=1}^{M} \boldsymbol{y}_\ell \, . \tag{8}$$

Similarly as $g(\eta) = \left(\sum_{j=1}^{p} \eta_j^2\right) - 1$ we get $\nabla g = \eta$. By substituting $\nabla f = \lambda \nabla g$ and using $\|\eta\| = 1$ we obtain

$$\eta = \frac{\sum_{\ell=1}^{M} \boldsymbol{y}_\ell}{\left\|\sum_{\ell=1}^{M} \boldsymbol{y}_\ell\right\|}\,. \tag{9}$$

Thus, we have computed η that maximises distance squared in Y.

2.3 Constructing a basis

We have computed the vector η that maximises $\sum_\ell \langle \eta, \boldsymbol{y}_\ell \rangle$. To find a basis, we need to find the second, third and subsequent vectors which are orthogonal and in some way maximise the quantity $\sum_\ell \langle \eta, \boldsymbol{y}_\ell \rangle$. Let us first rename η as η_1. Taking η_1 as the first basis vector, we take the projection of vector $\boldsymbol{y}_\ell$ on to η_1 and remove these components from $\boldsymbol{y}_\ell$.

$$\boldsymbol{y}_{\ell_1} = \boldsymbol{y}_\ell - \langle \eta_1, \boldsymbol{y}_\ell \rangle \, \eta_1 \tag{10}$$

Thus $\boldsymbol{y}_{\ell_1} \perp \eta_1$. Let $Y_1 = \{\boldsymbol{y}_{\ell_1}\}_{\ell=1}^{M}$ be the set of remaining components of $\{\boldsymbol{y}_\ell\}_{\ell=1}^{M}$ after removing the projection on to η_1. Now we can compute η_2 which maximises $\sum_\ell \left\langle \eta, \boldsymbol{y}_{\ell_1} \right\rangle$ as

$$\eta_2 = \frac{\sum_{\ell=1}^{M} \boldsymbol{y}_{\ell_1}}{\left\|\sum_{\ell=1}^{M} \boldsymbol{y}_{\ell_1}\right\|} \tag{11}$$

Proceeding in this way we obtain

$$\boldsymbol{y}_{\ell_b} = \boldsymbol{y}_{\ell_{b-1}} - \left\langle \eta_b, \boldsymbol{y}_{\ell_{b-1}} \right\rangle \eta_b\,, \tag{12}$$

$$\text{and} \qquad \eta_{b+1} = \frac{\sum_{\ell=1}^{M} \boldsymbol{y}_{\ell_b}}{\left\|\sum_{\ell=1}^{M} \boldsymbol{y}_{\ell_b}\right\|}\,, \tag{13}$$

with $\boldsymbol{y}_{\ell_0} = \boldsymbol{y}_\ell$. The set $\Theta = \{\eta_1, \eta_2, \ldots, \eta_p\}$ gives a basis for Y with each η_{i+1} maximising $\sum_\ell \left\langle \eta, \boldsymbol{y}_{\ell_i} \right\rangle$. We note that this process is somewhat similar to Gram Schmidt orthogonalisation.

Next we draw the reader's attention to a technical point. Equation (4) is only true for $\boldsymbol{y}_\ell \in \mathbb{R}^{p+}$. The other Y spaces $Y_1, Y_2, \ldots$ are not subsets of $\mathbb{R}^{p+}$. To adjust for this, after computing $\boldsymbol{y}_{\ell_b}$ as in equation (12) we temporarily change the basis such that the points $\{\boldsymbol{y}_{\ell_b}\}_{\ell=1}^{M}$ are in a positive orthant. Then after computing η_{b+1} as in equation (13) using positive $\{\boldsymbol{y}_{\ell_b}\}_{\ell=1}^{M}$, we transform it back again to the original basis.

Once the basis Θ is computed, we change the basis of the original data as follows:

$$\tilde{X} = X_1\Theta \quad \text{or} \quad \tilde{X} = X_2\Theta \tag{14}$$

where X_1 or X_2 is the normalized X space according to the choice of normalization. We note that the basis Θ is not invariant to rotations; i.e. if one were to start with a different basis, the resulting Θ would be different. This is because when constructing the Y space we compute element-wise squares of vector differences. While vector addition is basis invariant, squared element-wise components depend on the basis vectors. Notwithstanding this limitation, we achieve good results for experiments conducted on our extensive collection of more than 12,000 real world datasets. For almost all of our experiments, we constructed DOBIN using the original basis as input. We only used a different basis if the number of variables were much higher than the number of observations. Furthermore, as we will see in Section 4 it is easier to gain insights about the outlying directions when the axes are original variables.

We conclude this mathematical discussion by summarizing the key steps of DOBIN:

1. Find the Y space for a given dataset X as detailed in Algorithm 1.
2. Construct the basis Θ as outlined in Section 2.3.
3. Transform the original space X using equation (14).

Next we briefly consider the parameters involved in DOBIN.

2.4 Parameters of DOBIN

We have three numerical parameters k_1, k_2 and q for constructing the Y space with the following default values:

$$\begin{aligned} k_2 &= \min\left(20, \max\left(\lfloor N/20 \rfloor, 2\right)\right), \\ k_1 &= \max\left(k_2 - 10, 1\right), \\ q &= 0.95, \end{aligned} \tag{15}$$

where N is the number of observations in the dataset. The parameters k_1 and k_2 can increase the run time of the algorithm. Consequently we have a maximum value of 20 for k_2. As $k_1 < k_2$, we consider a maximum of 10 neighbours for each point. The parameter q represents a cut-off quantile needed to obtain a subset of $\boldsymbol{y}$ with large distances. Therefore q needs to be a relatively high value in the interval $(0, 1)$. From equation (5) we maximise the projection of $\boldsymbol{y}$ on to η. If q is too low, many $\boldsymbol{y}$ with smaller projections may contribute heavily to the sum, making DOBIN vectors less effective.

The other parameter for constructing the Y space is the choice of normalization. While Min-Max normalization is commonly used by the computer science community (Campos et al. 2016), other normalization techniques such as distance from univariate medians and Mahalanobis distance are commonly used by the statistical community (Billor et al. 2000). We give two options for normalization: Min-Max and Median-IQR.

The purpose of the final parameter is to aid visualisation in two dimensions. As points $\{\boldsymbol{y}_\ell\}_{i=1}^{M}$ lie in the positive orthant, the first vector η_1 which maximizes $\sum_\ell \langle \eta\, \boldsymbol{y}_\ell \rangle$ also lies in the positive orthant. However, this may not render the best possible 2D-visualization of the outliers in X space, because points can have positive and negative coordinates in X space. Hence we compute the signed $\boldsymbol{y}_\ell$, denoted by $\tilde{\boldsymbol{y}}_\ell$, by keeping track of the signs of the point differences $\left(\boldsymbol{x}_i - \boldsymbol{x}_j\right)$. Let $T(\ell) = (i, j)$, then

$$\tilde{\boldsymbol{y}}_\ell = \operatorname{sign}\left(\boldsymbol{x}_i - \boldsymbol{x}_j\right) \circ \boldsymbol{y}_\ell, \tag{16}$$

where $\circ$ denotes element-wise multiplication. Then

$$\begin{aligned} \operatorname{sign}(\eta_1) &= \operatorname{sign}\left(\sum_{i=1}^{M} \tilde{\boldsymbol{y}}_\ell\right), \\ \tilde{\eta}_1 &= \operatorname{sign}(\eta_1) \circ \eta_1, \end{aligned} \tag{17}$$

where η_1 is computed using equation (9). However, this adds to the computation and is only recommended if 2D-visualization is an aim of the exercise.

3 Experiments with synthetic data

In this section we do three experiments with synthetic data. For each experiment we compare results for three outlier detection methods, namely LOF (Breunig et al. 2000), KNN (Ramaswamy et al. 2000) and isolation forest (**?**). We choose these methods as they are well recognised and fundamentally different: LOF uses a local density based approach; KNN uses a global distance based approach; and iForest uses a randomised tree based approach. For each outlier detection method we consider the following three coordinate systems:

1. All variables in the dataset
2. Perform Principal Component Analysis (PCA) on the dataset, and use the first half of the principle components (PCs).
3. Perform DOBIN on the dataset, and use the first half of the DOBIN components.

For each outlier detection method, we compare the results using these three sets of coordinates. We use area under the Receiver Operator Characteristic curve (AUC) as our performance measure.

3.1 Experiment 1

The first experiment considers two normal distributions: an inlier normal distribution and an outlier normal distribution, which moves out from the inlier distribution as its mean increases. We consider a dataset of 405 observations

in $\mathbb{R}^6$, of which 400 observations in each dimension are normally distributed with mean 0 and standard deviation 1. The remaining 5 observations signify outliers and are normally distributed with mean μ and standard deviation 0.2 in one dimension, and mean 0 and standard deviation 1 in other dimensions. The value of μ is changed from 0 to 4.5 by increments of 0.5. The reason for a smaller standard deviation in the first dimension is to ensure that the outliers stay close to the mean μ and move out of the inlier distribution as μ increases. For each value of μ we consider the three sets of coordinates described above and their performance using the outlier methods LOF, KNN and iForest. We perform 10 iterations of this experiment. Figure 2 shows the results for all three outlier methods using the average performance values of 10 iterations.

Using 3 DOBIN components gives the highest performance for all 3 outlier detection methods for $\mu > 1$. Indeed, we are interested in a set of coordinates that can accentuate the outliers as they move out from the inlier distribution. For values of $\mu \in \{0, 0.5, 1\}$, the outlier distribution lies in the interior of the inlier distribution, making a performance comparison between the coordinates not meaningful.

3.2 Experiment 2

The second experiment considers a bi-modal setting with two inlier normal distributions and one outlier normal distribution which moves into the valley as its mean changes. Somewhat similar to the previous example we consider 805 observations in $\mathbb{R}^6$; 800 inlier observations of which 400 centred at $(5, 0, 0, 0, 0, 0)$ and the other 400 centred at $(-5, 0, 0, 0, 0, 0)$. All inlier observations come from normal distributions with standard deviation 1 in each dimension. The outlier distribution consists of 5 points with mean $(5 - \mu, 0, 0, 0, 0, 0)$ and standard deviations 0.2 in the first dimension and 1 in other dimensions. The value of μ is increased from 0 to 4.5 in increments of 0.5. Again, a smaller standard deviation is to ensure that the outliers move into the valley. Figure 3 shows the results for all three outlier methods on all three coordinate systems using the average values of 10 iterations.

Again we see that using 3 DOBIN components gives the highest performance for all three outlier methods.

3.3 Experiment 3

The third experiment is motivated from Zimek et al. (2012). We consider a uniform distribution in $\mathbb{R}^{20}$ and place a manual outlier at 0.9 in i dimensions where i is changed from 1 to 20. While this is not an outlier in any single dimension, as i increases this point stands out from the rest of the distribution. Indeed, for large i this point lies furthest from the rest. An equivalent outlier is 0.1 in i dimensions. The reason that these points are outliers is because they are at the corners of the hyper-cube and as such are far away from other points. On the contrary, 0.5 in i dimensions would be closest to other points. This is because $(0.5, 0.5, \ldots)$ has neighbours from all sides and has on average 2^{20} more neighbours compared to $(1, 1, \ldots)$ which is a corner on the unit hyper-cube. For practical purposes the point $(0.9, 0.9, \ldots)$ is similar to $(1, 1, \ldots)$ and has significantly less neighbours compared to $(0.5, 0.5, \ldots)$. Figure 4 shows the results of all three outlier methods on all three coordinate systems. Again we see that using half of the DOBIN components gives the best performance for all three outlier methods.

In all three experiments we see that DOBIN coordinates give the best performance followed by the full set of coordinates. Principal Components are generally less effective than the full set of coordinates for these three experiments and outlier methods. We note that this is not a failure of PCA, as outlier detection is not its intended goal. Rather, PCA maximizes the variance in each component such that a low dimensional representation of the dataset is a good approximation to the original. However as PCA is used in outlier detection when dimension reduction is preferred we compare DOBIN with PC components, and propose DOBIN as an alternative to PCA as a dimension reduction technique for outlier detection.

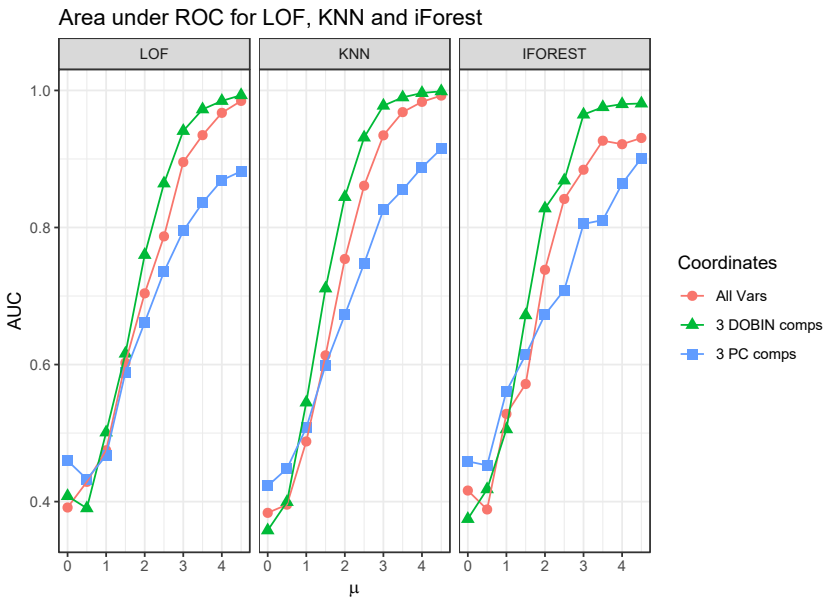


Figure 2: *Results of Experiment 1, in* $\mathbb{R}^6$.

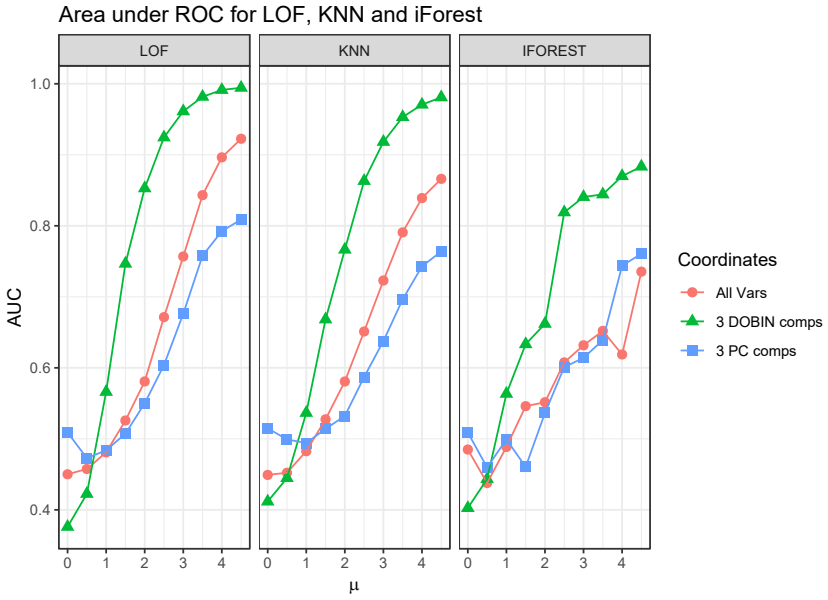


Figure 3: *Results of Experiment 2, in* $\mathbb{R}^6$.

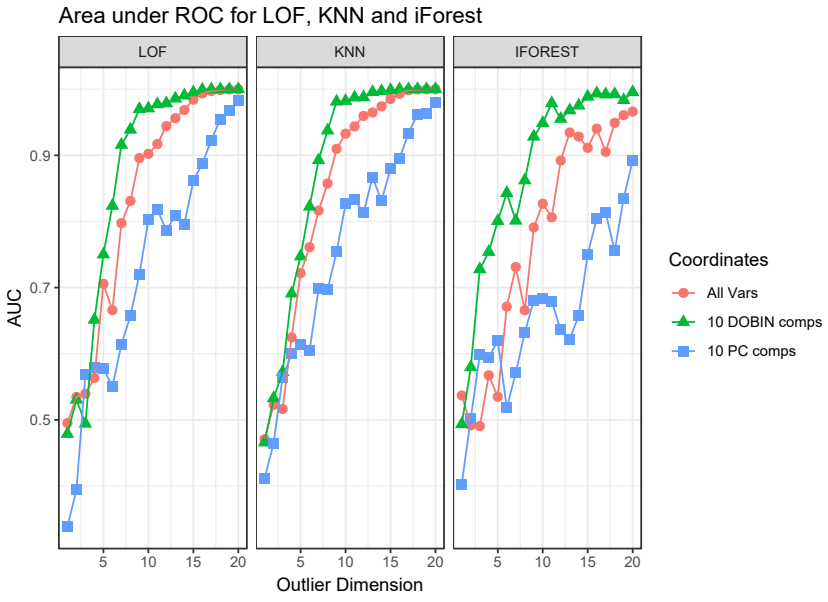


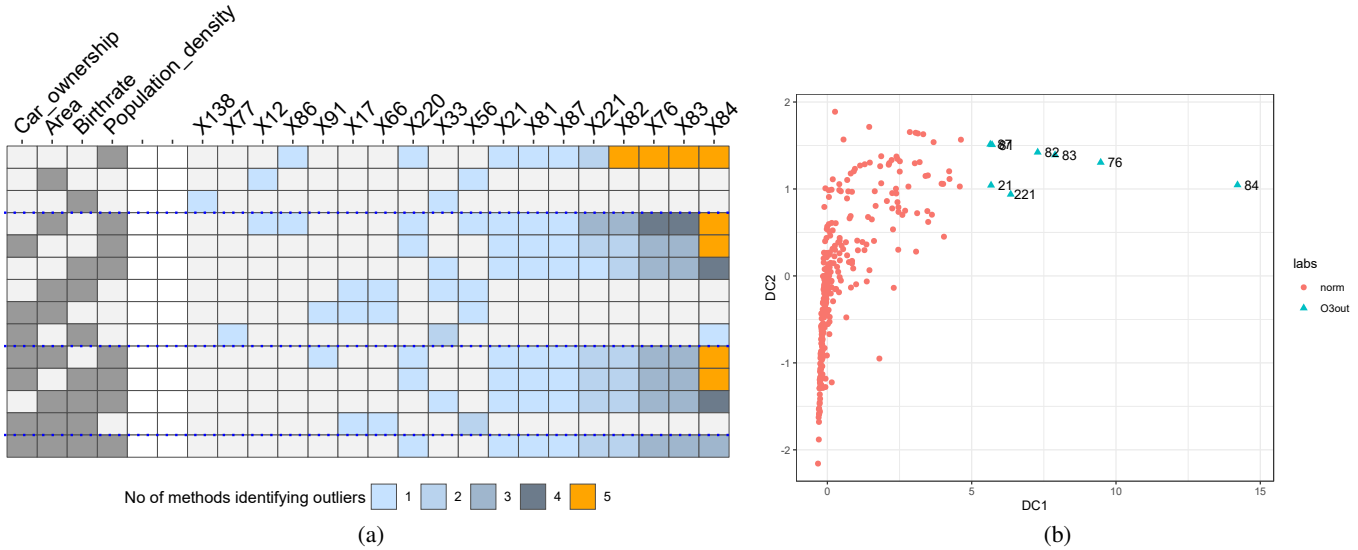
Figure 4: *Results of Experiment 3, in* $\mathbb{R}^{20}$.

4 Results with visualization

In this section we investigate specific examples motivated from Unwin (2019*a*) and Wilkinson (2017) using O3 plots and the DC1-DC2 space, which is the space spanned by the first two DOBIN vectors. O3 plots (Unwin 2019*a*) can be used to compare the results of 6 different outlier detection methods: *HDoutliers*, *mvBACON*, *adjOutlyingness*, *covMcd* , *FastPCs*, and *DetectDeviatingCells*. Besides facilitating an ensemble of outlier methods, O3 plots also highlight outliers in axis parallel subspaces. We use O3 plots as a means of validating the DC1-DC2 space.

4.1 *Election2005* dataset

We start with the *Election2005* dataset (Grimm 2019), which is used by Unwin (2019*a*) to illustrate the O3 plot. This dataset includes election results of two German elections and certain attributes of the electorates. We examine the O3 plot and the DC1-DC2 space of this dataset.



(a) (b)

Figure 5: *O3plot of the Election2005 dataset in Figure 5a and the first 2 DOBIN components in Figure 5b*

Figure 5a shows the O3 plot of this dataset using all six outlier methods. The columns in the left indicate the variables, the columns in the right indicate the observations, the rows specify the axis parallel subspaces and the colours depict the number of methods that identify each observation in each subspace as an outlier. From this plot we see that observation $X84$ is identified as an outlier by 5 methods in 5 subspaces, 4 methods in 2 subspaces, 3 methods in 1 subspace and by 1 method in 1 subspace. $X84$ is arguably the most outlying observation in this dataset. The observations $X83$, $X76$, $X82$ are also identified as outliers by 5 methods in the dimension of population density. They are also identified as outliers by multiple methods in different subspaces. The layout of the O3 plot is such that the outlyingness of the observations increase to the right.

Figure 5b shows the first 2 DOBIN components of the *Election2005* dataset. This is a projection of the dataset onto a two dimensional subspace spanned by the first 2 DOBIN vectors. Here we see the observation $X84$ (denoted by 84 on the plot) far away from the rest of the data with $X76$, $X83$ and $X82$ somewhat detached from the rest. The observations $X87$ an $X81$ are almost at the same position in this 2-dimensional space. The observations $X221$ and $X21$ also appear a bit outside the boundary of other points. A simple KNN algorithm in the DC1-DC2 space gives the top 8 candidates as $X84$, $X76$, $X83$, $X82$, $X221$, $X81$, $X87$, and $X21$. Thus, the DC1-DC2 space accentuates the eight most outlying observations according to the O3 plot, while showing clearly that $X84$ is the most outlying observation.

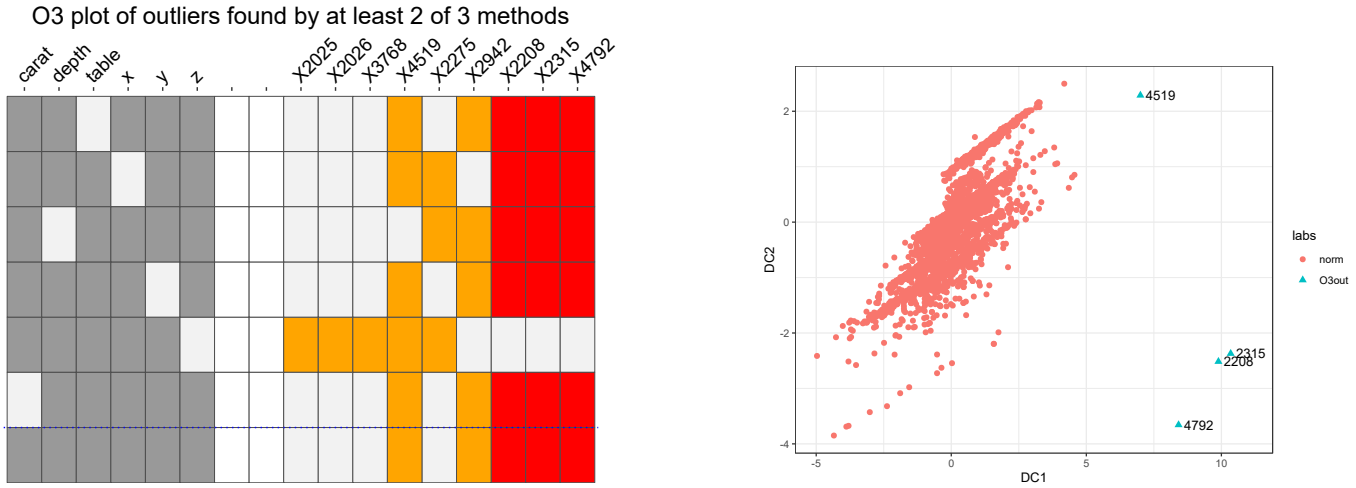
The first DOBIN vector is of interest to us as the outliers deviate in this direction. Equation (18) gives the first DOBIN vector, DC1 as a linear combination of the input variables.

$$\text{DC1} = \begin{bmatrix} 0.008 & 0.998 & 0.040 & -0.050 \end{bmatrix} \begin{bmatrix} \text{Area} \\ \text{Population Density} \\ \text{Birthrate} \\ \text{Car Ownership} \end{bmatrix} \tag{18}$$

From equation (18) we see that *Population Density* is the main variable contributing to outliers in the DC1-DC2 space. The O3 plot supports this observation. If we look carefully at the O3 plot we see that $X21$ to $X84$ gets identified as outliers in all subspaces that contain *Population Density*. This insight gives us a better understanding of the dataset.

4.2 *Diamonds* dataset

The next example is taken from the *O3Outliers* R package (Unwin 2019*b*) and uses the *Diamonds* dataset (Wickham 2016). This dataset contains attributes of diamonds.



(a) (b)

Figure 6: *O3plot of the Diamonds dataset in Figure 5a and the first 2 DOBIN components in Figure 5b.*

Figure 6a shows the O3 plot of this dataset using the three outlier methods *HDoutliers*, *FastPCs* and *adjOutlyingness*. We could not use all six methods on this dataset as the other three methods included in the *OutliersO3* R package gave computational errors. Of the three methods used, all methods identified $X4792$, $X2315$ and $X2208$ as outliers in 6 subspaces including the full space. These 3 points appear in the DC1-DC2 space in the bottom right corner. In addition, the point $X4519$ (denoted by 4519 in the plot) appear in the top right corner in the DC1-DC2 space, away from other points. Comparing with the O3 plot, we see that $X4519$ is identified as an outlier by 2 methods in 6 subspaces. In keeping with the guidelines of the previous section, we use half of the DC components to evaluate a KNN algorithm. The top 4 candidates in this 3-dimensional DC space are $X4792$, $X2315$, $X2208$ and $X4519$ in that order.

To understand the first DOBIN vector, we look at its coefficients:

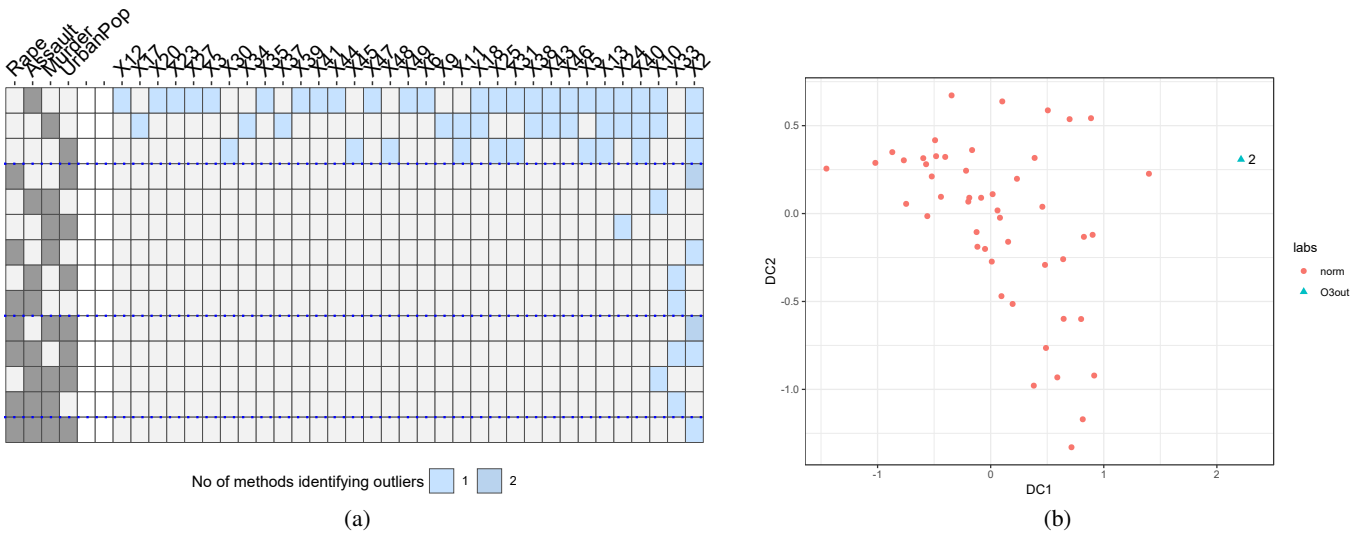
$$\text{DC1} = \begin{bmatrix} 0.098 & -0.659 & 0.185 & 0.097 & 0.126 & -0.704 \end{bmatrix} \begin{bmatrix} \text{Carat} \\ \text{Depth} \\ \text{Table} \\ \text{x} \\ \text{y} \\ \text{z} \end{bmatrix} \tag{19}$$

We see that the variables z and *Depth* contribute more to outliers in the DC1-DC2 space as well as in the O3 plot.

4.3 *USArrests* dataset

The popular *USArrests* dataset is studied by Bailey & Gatrell (1995), Sarkar (2008), Yamini (2019) to name a few. This dataset contains violent crime rates for the 50 US states in 1973. The O3 plot using all six outlier detection methods and the DC1-DC2 space of this dataset are illustrated in Figure 7.

From Figure7a we see that $X2$ is flagged as an outlier in 8 subspaces including the full space, by one or two outlier detection methods. The adjoining outlier $X33$ is only flagged in 4 subspaces and the others in even smaller subspaces. The observation $X2$ corresponds to Alaska. Of the six outlier methods, only two identify Alaska as an outlier in certain subspaces. This also suggests that even though there is consensus between two methods, as only a third of the



(a) (b)

Figure 7: *O3plot of the USArrests dataset in Figure 7a and the first 2 DOBIN components in Figure 7b.*

methods agree on this outlier, it may not be quite so outlying as the outliers in the previous examples. Indeed, this notion is validated by the DC1-DC2 plot. We see the observation $X2$ (denoted by 2 in the plot) standing out from the rest, but not by a massive margin. A KNN algorithm identifies this observation as having the highest KNN distance in DC1-DC2 space.

We look at the coefficients of the first DOBIN vector as $X2$ deviates in DC1:

$$\text{DC1} = \begin{bmatrix} 0.173 & 0.077 & -0.675 & 0.713 \end{bmatrix} \begin{bmatrix} \text{Murder} \\ \text{Assault} \\ \text{UrbanPop} \\ \text{Rape} \end{bmatrix} \tag{20}$$

We see that the variables *UrbanPop* and *Rape* contribute more to the first DOBIN vector than the other variables. From equation (20), high values of DC1 indicate low percentages of urban population and high rape arrests per 100,000 residents. As such, we might expect Alaska to have a higher rape arrests per urban population percentage. Indeed from Figure 8 we see that Alaska has the highest ratio of *Rape/UrbanPop* of all 50 states confirming the insight gleaned from the first DOBIN vector.

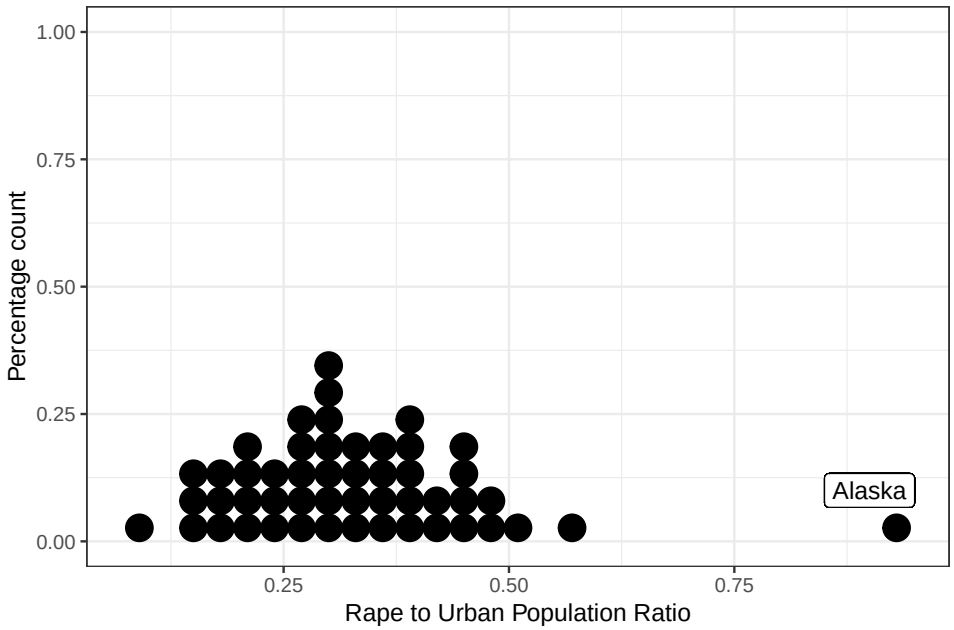
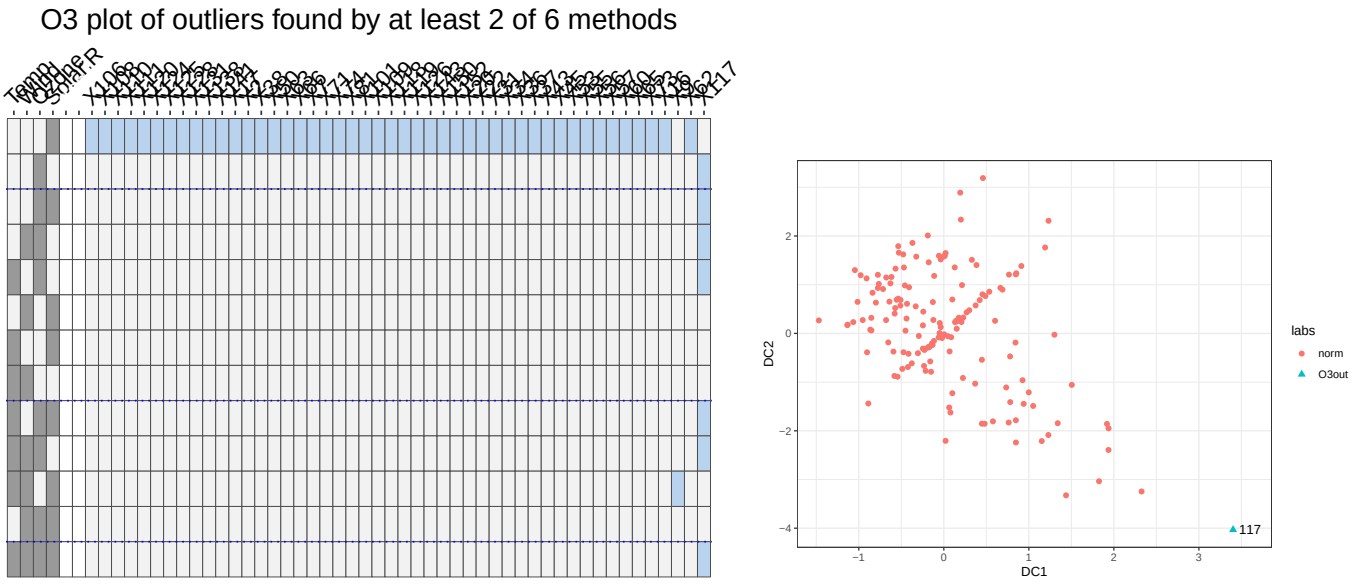


Figure 8: *USArrests dataset – Rape to Urban population ratio values.*

4.4 *Airquality* dataset

The *Airquality* dataset discussed in John et al. (1983) has measurements on air quality in New York city from May to September 1973. The O3 plot using all 6 outlier methods and the DC1-DC2 space of this dataset are shown in Figure 9.



(a) (b)

Figure 9: *O3plot of the Airquality dataset in Figure 9a and the first 2 DOBIN components in Figure 9b.*

The O3 plot in Figure 9a shows $X117$ as the only outlier identified in 6 subspaces by 2 methods. The associated DC1-DC2 space shows the observation $X117$ (denoted by 117 in the plot) away from the rest at the bottom right. Again, we see agreement between the O3 plot and DC1-DC2 space. The observation $X117$ was identified as an outlier only by a third of the outlier methods. This is corroborated in the DC1-DC2 plot by comparatively less outlyingness of observation $X117$ compared to outliers in the *Diamonds* dataset. Performing a KNN algorithm revealed the observation with the highest KNN distance as $X117$ in the DC1-DC2 space.

The first DOBIN vector has the following coefficients:

$$\text{DC1} = \begin{bmatrix} 0.842 & -0.078 & 0.512 & -0.151 \end{bmatrix} \begin{bmatrix} \text{Ozone} \\ \text{Solar.R} \\ \text{Wind} \\ \text{Temp} \end{bmatrix} \tag{21}$$

From equation (21) we see that the observation $X117$ is an outlier due to high values of *Ozone* and *Wind*.

4.5 dataset

The dataset (Vialaneix et al. 2019) contains the character coappearance network of characters in the novel *Misérables* by Victor Hugo. This dataset is stored as a graph, with vertices corresponding to characters and is shown in Figure 10. The character network in *Misérables* is also studied in Wilkinson (2017) focusing on outliers. Here we conduct a similar study using the O3 plot and the DC1-DC2 space.

As in Wilkinson (2017) we compute the graph-based features centrality, transitivity, closeness, betweenness, degree, average nearest neighbour degree and page rank for each character in the dataset. This transforms the original graph to a rectangular dataset of 77 observations and 7 variables, with each observation corresponding to a character in the novel. Figure 11a shows the O3 plot for the transformed dataset using the three methods *HDoutliers*, *mvBACON* and *covMCD*. The other three outlier methods used in O3 plots gave computational errors and as such could not be used. Figure 11b shows the DC1-DC2 space of the transformed dataset.

From the O3 plot it is evident that Valjean is the main outlier consistently identified by the three outlier methods in a vast majority of the subspaces. This is confirmed by the DC1-DC2 plot as Valjean appears far away from the rest of

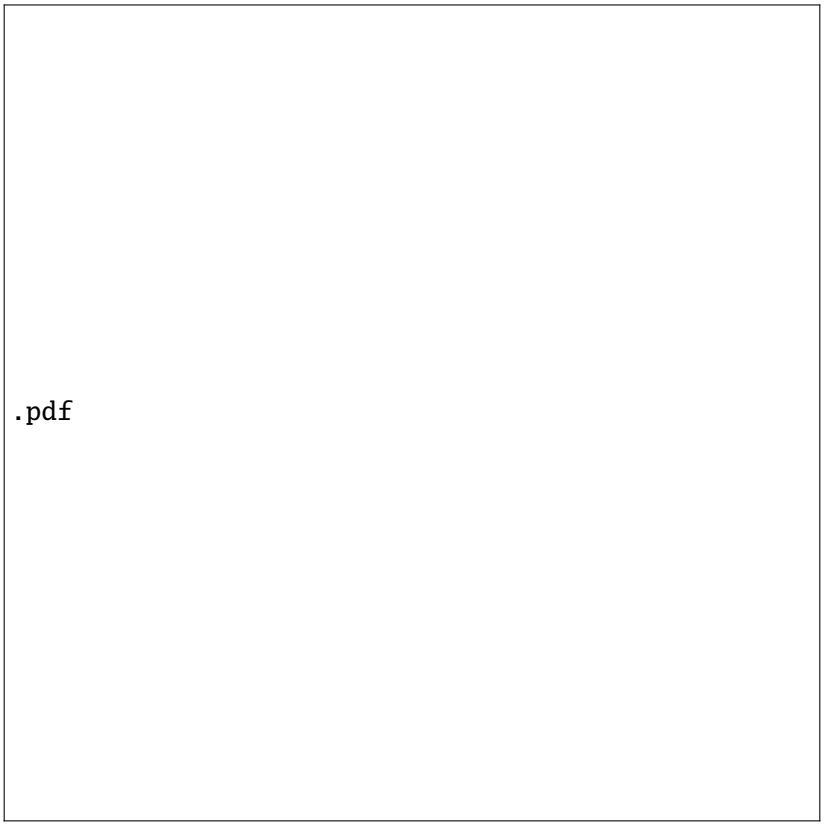
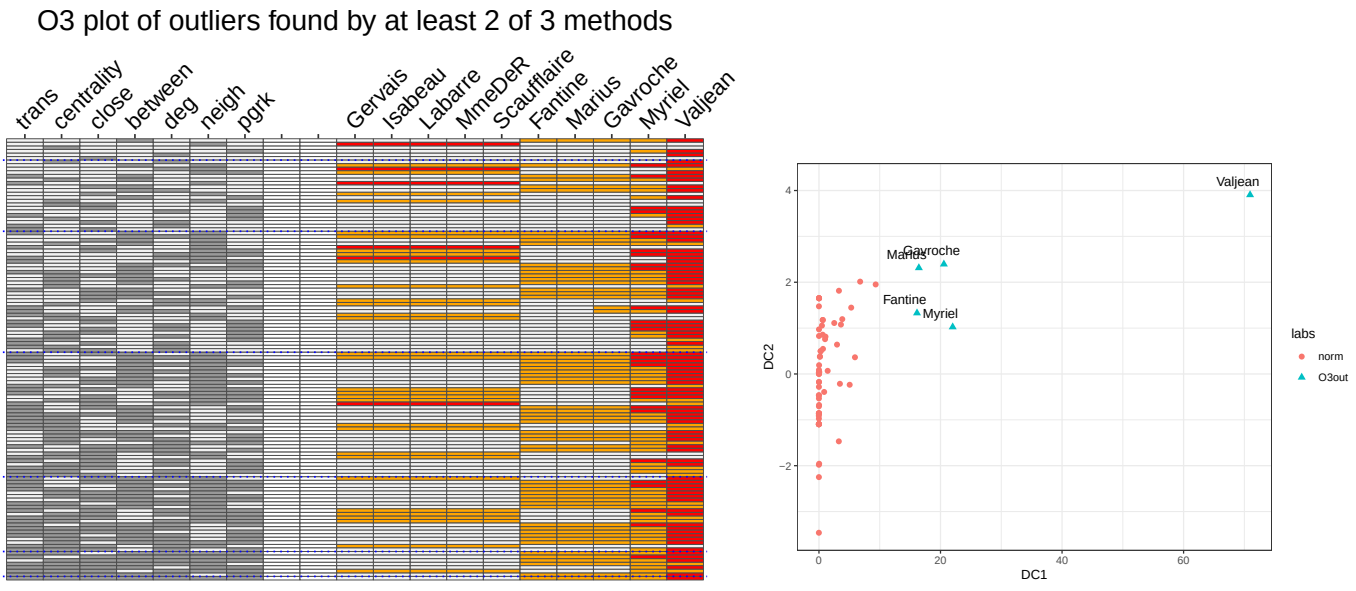


Figure 10: *The dataset.*



(a) (b)

Figure 11: *O3plot of the transformed dataset in Figure 11a and the first 2 DOBIN components in Figure 11b.*

the characters in the top right corner. The next most outlying characters in the O3 plot are Myriel, Gavroche, Marius and Fantine. These characters also appear in the DC1-DC2 space somewhat detached from the rest of the characters, although none so remarkably far away as Valjean. A KNN algorithm performed on the first half of the DC components identified Valjean, Myriel, Gavroche, Marius and Fantine as having the highest KNN distances.

As the first DOBIN vector is discriminatory, we look at its coefficients: The first DOBIN vector has the following

coefficients:

$$\text{DC1} = \begin{bmatrix} 0.0022 & -0.0001 & 0.0032 & 0.9999 & 0.0029 & 0.0025 & 0.0057 \end{bmatrix} \begin{bmatrix} \text{Centrality} \\ \text{Transitivity} \\ \text{Closeness} \\ \text{Betweenness} \\ \text{Degree} \\ \text{Avg. Neighbour Degree} \\ \text{Page Rank} \end{bmatrix}. \quad (22)$$

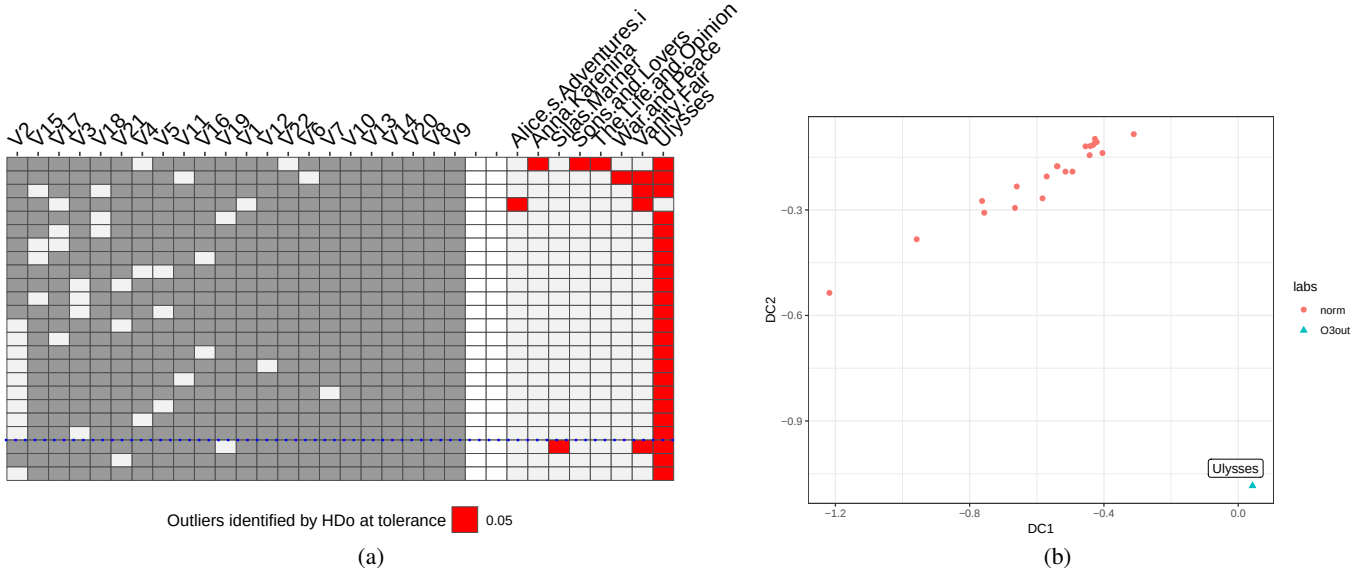
From equation (22) we see that *Betweenness* is the main feature that makes Valjean an outlier in the DC1-DC2 space. The property *Betweenness* is defined as the number of shortest paths going through a vertex (Csardi & Nepusz 2006). Thus Valjean becomes a node in many shortest paths in the character network making him the main outlier.

4.6 Classics from Gutenberg

This example is on text analysis of 22 classics downloaded from the Gutenberg project (*Project Gutenberg* n.d.) and is similar to the example in Wilkinson (2017). We consider the novels *Alice in Wonderland*, *Anna Karenina*, *Bleak House*, *Emma*, *Frankenstein*, *Gullivers Travels*, *Jude the Obscure*, *Jim*, *Mansfield Park*, *Middlemarch*, *Moby Dick*, *Northanger Abbey*, *Persuasion*, *Pride and Prejudice*, *Sense and Sensibility*, *Silas Marner*, *Sons and Lovers*, *The Life and Opinions of Tristram Shandy*, *Wizard of Oz*, *Ulysses*, *Vanity Fair* and *War and Peace* in our analysis.

We strip each novel into words and obtain the collection of words used and their relative frequencies. This set of words need to be cleaned before computing any useful features. We use the R package *tidytext* (Silge & Robinson 2016) for this task. As the first 'cleaning' step, we eliminate the stop words such as *the*, *a*, *an* and *in* from our collection. We also remove numbers from our word collection. Next we use a stemming procedure to reduce words to their word stem. For example the words *run* and *running* have the same word stem. This process gives us a cleaned set of words for each novel. We compute the tf-idf (term frequency-inverse document frequency) measure on this collection. The tf-idf statistic measures how important a word is to a document when considering a collection of documents. Consequently we end up with a rectangular dataset of 22 observations and 39,624 variables, where each observation is a novel and each variable is the tf-idf statistic of a word.

However, 22 vectors in $\mathbb{R}^{39,624}$ can only span a subspace of 22 dimensions or less. As such, we compute this subspace and the respective coordinates of the observations using PCA. We use all the principal components so that we're performing a change of basis and not a projection. As a result we obtain a 22×22 matrix, where each row is an observation corresponding to a novel and each column is a linear combination of 39,624 tf-idf statistics. The resulting O3 plot using *HDoutliers* and the corresponding DC1-DC2 space is illustrated in Figure 12. Again we see the O3 plot



(a) (b)

Figure 12: *O3plot of 22 Gutenberg classics in Figure 12a and the first 2 DOBIN components in Figure 12b.*

and the DC1-DC2 space agreeing on the most outlying novel *Ulysses* confirming the findings of Wilkinson (2017). Performing a KNN algorithm yielded *Ulysses* as the observation with the highest KNN distance.

The novel *Ulysses* stands out in the DC1-DC2 space mainly because of its second DC component. However, we do not explore the coefficients of this vector as each coefficient is a linear combination of $39,624$ variables, making it difficult to gain insights.

This section substantiated DOBIN's ability as an outlier visualization tool. In each example we saw the outliers projected onto a 2D space spanned by the first two DOBIN vectors, and their legitimacy was endorsed by the O3 plots. Furthermore, we gained insights about the nature of outlyingness in data by inspecting DOBIN vectors.

This completes the visual analysis of datasets. Next we explore the effect of DOBIN on an extensive data repository.

5 Results on a data repository

In this section we work with a data repository of $12,642$ datasets (Kandanaarachchi et al. 2019), generated from approximately 200 source datasets, which was used in our earlier work. Each dataset has labeled outliers and originally was used for classification purposes. The present datasets are modified from its original form by downsampling minority class observations, converting categorical attributes to numerical values, removing missing data and creating several variants from each classification dataset. More details on dataset preparation can be found in Kandanaarachchi et al. (2018). This collection also includes datasets from Campos et al. (2016).

From an application view point, it is fortunate if outliers arise in the full input space. It is more likely that outliers arise in subspaces, and is equally important that these are detected in a timely manner. If detecting subspace outliers is a goal, using downsampled classification datasets for outlier detection is sub-optimal. This is because classification datasets generally contain variables that are needed for effective classification, and as a result the outliers in the downsampled dataset may arise in the full space. However, as downsampled classification datasets have labeled outliers, they enable effective evaluation of different pre-processing and outlier methods. Thus, to make these datasets suitable for subspace outlier detection, we "fatten" each dataset by adding 20 noise variables distributed normally, i.e. $\sim \mathcal{N}(0,1)$. We use these 'fattened' datasets in our experiment.

Similar to Section 3, for each dataset we compare the results of three outlier methods LOF, KNN and iForest using the area under the ROC curve (AUC), on three different coordinate systems, namely the full set of coordinates, the first half of DOBIN components and the first half of Principal Components. For a dataset with p variables and N observations, if $p > N$, we consider $N/2$ number of DOBIN and PC components. This is because when $p > N$, N vectors can only span an N dimensional space, which is smaller than a p dimensional space.

Furthermore, we need to select appropriate tools to present results of $12,642$ datasets across three methods and three coordinate systems. The ability to delve into each dataset and obtain insights is no longer feasible with a large repository. In addition, we want to ascertain whether DOBIN coordinates improve the performance of the three outlier detection methods compared to the other two coordinate systems. Therefore, we investigate each outlier detection method separately. For each outlier method, we compare the performance of the three coordinate systems using a Friedman test adjusting for the dataset variants. To adjust for the dataset variants, we take the median performance for each dataset source and each coordinate system. If the Friedman test is significant, then we perform a Nemenyi test, so that we can rank the coordinate systems according to performance. For convenience we use the shortened names $1/2$ DOBIN for using the first half of DOBIN components, $1/2$ PCA for using the first half of Principal Components and 'All Vars' for using all variables in the remainder of the section.

5.1 Significant tests

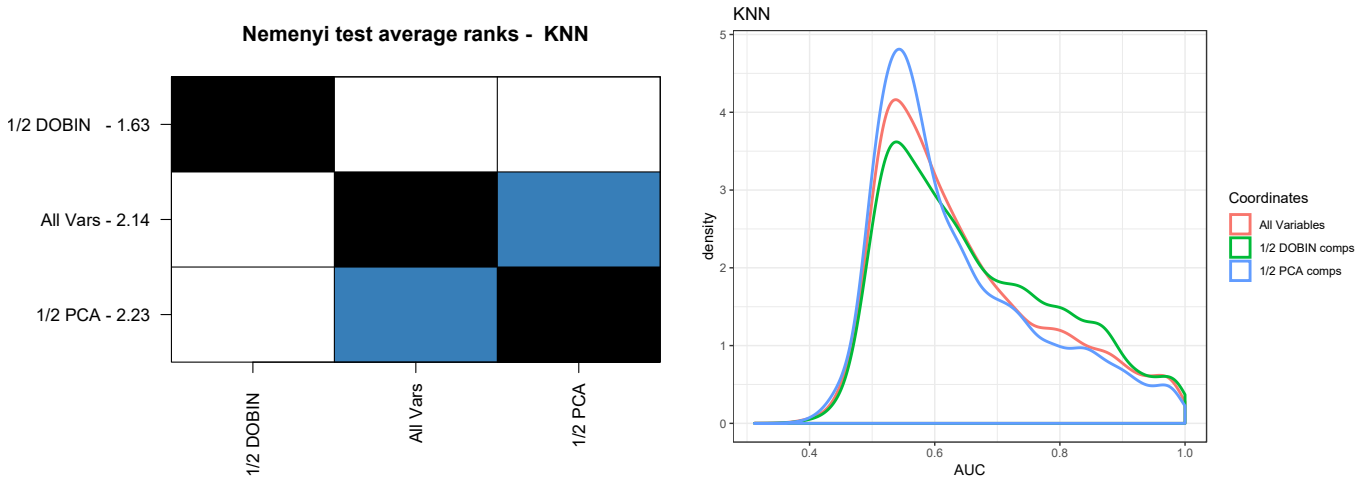
Figure 13 gives the results of the three coordinate systems on outlier detection method LOF. A Friedman test gave the p-value 1.986×10^{-5}, and Figure 13a shows the associated Nemenyi plot for this data. This plot shows the ranks of the coordinate systems, with lower ranks indicating better performance. The blue squares highlight methods which are not significantly different from each other. From Figure 13a we see that the best coordinates for LOF are $1/2$ DOBIN, followed by $1/2$ PCA and All Vars, with $1/2$ PCA and All Vars not significantly different from each other. Figure 13b shows the distribution of performance results for LOF using the three coordinate systems. As these curves represent probability density estimates, the area under each curve is 1. We see that $1/2$ DOBIN components give rise

to larger density values for AUC > 0.7, i.e. it has a bigger proportion of high AUC values compared to the other two coordinate systems.

(a) (b)

Figure 13: *Nemenyi plot of LOF results on the data repository in Figure 13a and the probability density plot of the three coordinate systems in Figure 13b.*

Figure 14 shows the results of the three coordinate systems on the outlier detection method KNN. We obtained the p-value 1.341×10^{-8} from the Friedman test. Again, from Figure 14a we see that $1/2$ DOBIN outperforms All Vars and $1/2$ PCA, which is corroborated by the probability density plot in Figure 14b. Even though All Vars is ranked second, it is not significantly different from $1/2$ PCA for KNN.



(a) (b)

Figure 14: *Nemenyi plot of KNN results on the data repository in Figure 14a and the probability density plot of the three coordinate systems in Figure 14b.*

Similar performances are obtained for iForest, which is illustrated in Figure 15. A Friedman test conducted on the data gave a p-value less than 2.2×10^{-16}. The Nemenyi plot shows that while $1/2$ DOBIN performs the best, all three methods are significantly different from each other, with All Vars outperforming $1/2$ PCA. This is supported from the probability density plot in Figure 15b, with the curve associated to $1/2$ DOBIN lying above the All Vars curve, and the All Vars curve lying above $1/2$ PCA curve for AUC > 0.65.

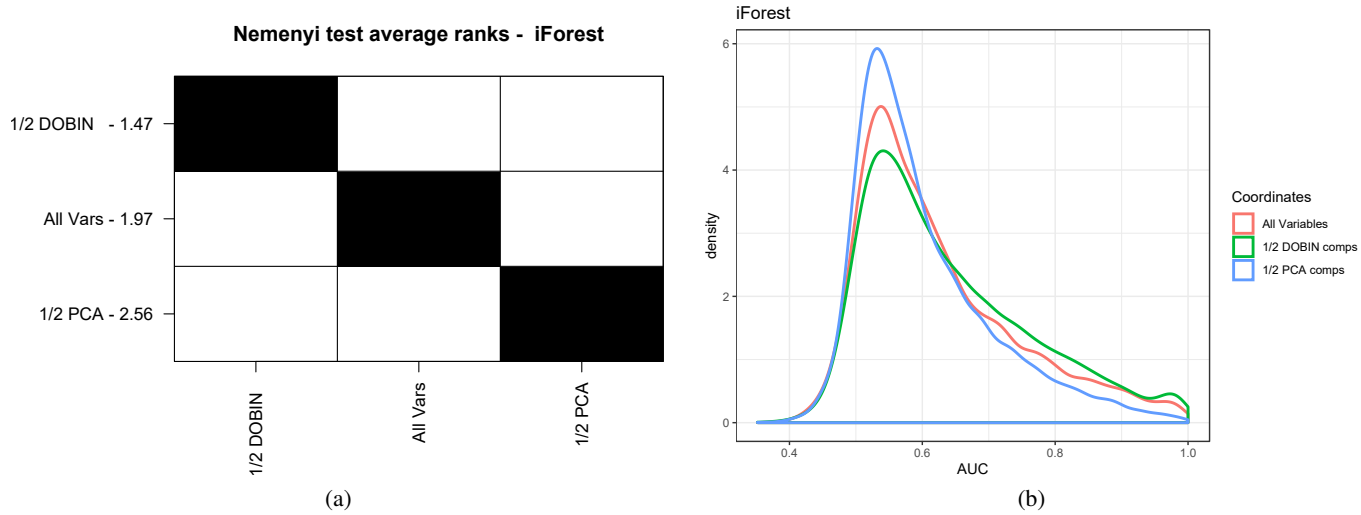


Figure 15: *Nemenyi plot of iForest results on the data repository in Figure 15a and the probability density plot of the three coordinate systems in Figure 15b.*

We see that 1/2 DOBIN performs best for all three outlier detection methods and is significantly better than 1/2 PCA and All Vars. This is evidence that DOBIN is better as a dimension reduction tool for outier detection compared to PCA and the original variables. We attribute this success to DOBIN's firm mathematical foundations.

6 Conclusion

Dimension reduction for outlier detection is a topic that has not received much attention in the literature. One way to achieve this is to construct a basis that empowers outlier detection. In this paper, we present DOBIN: a Distance based Outlier Basis usIng Neighbours, which strives to accomplish this goal. We used DOBIN on an extensive data repository of real world datasets and showed that DOBIN basis improves performance of the outlier detection methods LOF, KNN and iForest. Using half of the DOBIN components we outperformed PCA and the standard basis as coordinate systems for these three outlier detection methods.

In addition, we conducted a visual analysis of outliers using DOBIN on six datasets from diverse sources, which included a character graph from the novel *Misérables* and 22 classics from Gutenberg website. We projected the data onto a 2D space spanned by the first two DOBIN vectors, and saw that outliers identified by many methods in O3 plots were indeed far away from the rest of the data. We gained insights about the outliers in these datasets, and identified the variables that contributed to make them outliers.

Currently DOBIN is sensitive to rotations in the input space. As future work we plan to investigate avenues that can contribute to a rotation invariant version of DOBIN. Such a version may be advantageous for working with datasets that are already pre-processed. The R package *dobin* contains the implementation and is available at `https://github.com/sevvandi/dobin` for download.

Supplementary Material

R package dobin: This package contains the DOBIN basis construction.

Datasets: Datasets referred in Section 5 are available at Kandanaarachchi et al. (2019). The character network from the novel *Misérables* was taken from the R package *SOMbrero* (Vialaneix et al. 2019). The book vectors dataset used in Section 4.6 is available at the github repository `https://github.com/sevvandi/Outlier-Basis`.

Scripts: The script `Figures_For_Paper_1.R` contains the R code used to conduct experiments and produce graphs in Section 3. The script `Figures_For_Paper_2.R` contains the R code used in Section 4. And finally, the script `Figures_For_Paper_3.R` contains the R code used to produce the graphs in Section 5. The original computation on the data repository detailed in Section 5 was conducted using the Monarch HPC cluster. The results of this computation are available at the github repository `https://github.com/sevvandi/Outlier-Basis`.

Other R-packages: Within the R package *dobin* we have used *pracma* (Borchers 2019) and *FNN* (Beygelzimer et al. 2018). In addition, we have used the R packages *OutliersO3* (Unwin 2019*b*), *isolationForest* (Liu 2009), *DMwR* (Torgo 2010), *pROC* (Robin et al. 2011), *graphics* (R Core Team 2018), *ggplot2* (Wickham 2016), *mbgraphic* (Grimm 2019), *gridExtra* (Auguie 2017), *tidyverse* (Wickham 2017) and *reshape2* (Wickham 2007).

Acknowledgements

Funding was provided by the Australian Research Council through the Linkage Project LP160101885. This research was also supported by the Monash eResearch Centre and eSolutions-Research Support Services through the MonARCH HPC Cluster.

References

Aggarwal, C. C. & Yu, P. S. (2001), Outlier detection for high dimensional data, *in* 'ACM Sigmod Record', Vol. 30, ACM, pp. 37–46.

Auguie, B. (2017), *gridExtra: Miscellaneous Functions for "Grid" Graphics*. R package version 2.3.
URL: *https://CRAN.R-project.org/package=gridExtra*

Bailey, T. C. & Gatrell, A. C. (1995), *Interactive spatial data analysis*, Vol. 413, Longman Scientific & Technical Essex.

Beygelzimer, A., Kakadet, S., Langford, J., Arya, S., Mount, D. & Li, S. (2018), *FNN: Fast Nearest Neighbor Search Algorithms and Applications*. R package version 1.1.2.2.
URL: *https://CRAN.R-project.org/package=FNN*

Billor, N., Hadi, A. S. & Velleman, P. F. (2000), 'Bacon: blocked adaptive computationally efficient outlier nominators', *Computational statistics & data analysis* **34**(3), 279–298.

Borchers, H. W. (2019), *pracma: Practical Numerical Math Functions*. R package version 2.2.5.
URL: *https://CRAN.R-project.org/package=pracma*

Breunig, M. M., Kriegel, H.-P., Ng, R. T. & Sander, J. (2000), LOF: identifying density-based local outliers, *in* 'ACM Sigmod Record', Vol. 29, ACM, pp. 93–104.

Brys, G., Hubert, M. & Rousseeuw, P. (2005), 'A robustification of independent component analysis', *Journal of Chemometrics: A Journal of the Chemometrics Society* **19**(5-7), 364–375.

Campos, G. O., Zimek, A., Sander, J., Campello, R. J., Micenková, B., Schubert, E., Assent, I. & Houle, M. E. (2016), 'On the evaluation of unsupervised outlier detection: measures, datasets, and an empirical study', *Data Mining and Knowledge Discovery* **30**(4), 891–927.

Csardi, G. & Nepusz, T. (2006), 'The igraph software package for complex network research', *InterJournal* **Complex Systems**, 1695.
URL: *http://igraph.org*

Goldstein, M. & Uchida, S. (2016), 'A comparative evaluation of unsupervised anomaly detection algorithms for multivariate data', *PloS one* **11**(4), e0152173.

Grimm, K. (2019), *mbgraphic: Measure Based Graphic Selection*. R package version 1.0.1.
URL: *https://CRAN.R-project.org/package=mbgraphic*

Hyndman, R. J., Wang, E. & Laptev, N. (2015), Large-scale unusual time series detection, *in* '2015 IEEE international conference on data mining workshop (ICDMW)', IEEE, pp. 1616–1619.

John, C., Cleveland, W., Kleiner, B. & Tukey, P. (1983), 'Graphical methods for data analysis', *Pacific Grove: Wadsworth* pp. 158–62.

Kandanaarachchi, S. (2019), *dobin: Dimension Reduction for Outlier Detection*. R package version 0.0.0.9000.
URL: *https://github.com/sevvandi/dobin*

Kandanaarachchi, S., Muñoz Acosta, M., Smith-Miles, K. & Hyndman, R. (2019), 'Datasets for outlier detection'.
URL: *https://monash.figshare.com/articles/Datasets_12338_zip/7705127/4*

Kandanaarachchi, S., Munoz, M. A., Hyndman, R. J., Smith-Miles, K. et al. (2018), 'On normalization and algorithm selection for unsupervised outlier detection'. pre-print.
URL: *http://bit.ly/normout*

Keller, F., Muller, E. & Bohm, K. (2012), Hics: High contrast subspaces for density-based outlier ranking, *in* '2012 IEEE 28th international conference on data engineering', IEEE, pp. 1037–1048.

Kriegel, H.-P., Kröger, P., Schubert, E. & Zimek, A. (2009), Loop: local outlier probabilities, *in* 'Proceedings of the 18th ACM conference on Information and knowledge management', ACM, pp. 1649–1652.

Liu, F. T. (2009), *IsolationForest: Isolation Forest*. R package version 0.0-26/r4.
URL: *https://R-Forge.R-project.org/projects/iforest/*

Liu, F. T., Ting, K. M. & Zhou, Z.-H. (2008), Isolation forest, *in* '2008 Eighth IEEE International Conference on Data Mining', IEEE, pp. 413–422.

Project Gutenberg (n.d.), `http://www.gutenberg.org`. Accessed: 2019-08-30.

R Core Team (2018), *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria.
URL: *https://www.R-project.org/*

Ramaswamy, S., Rastogi, R. & Shim, K. (2000), Efficient algorithms for mining outliers from large data sets, *in* 'ACM Sigmod Record', Vol. 29, ACM, pp. 427–438.

Robin, X., Turck, N., Hainard, A., Tiberti, N., Lisacek, F., Sanchez, J.-C. & Müller, M. (2011), 'proc: an open-source package for r and s+ to analyze and compare roc curves', *BMC Bioinformatics* **12**, 77.

Rousseeuw, P. J. & Bossche, W. V. D. (2018), 'Detecting deviating data cells', *Technometrics* **60**(2), 135–145.

Rousseeuw, P. J. & Driessen, K. V. (1999), 'A fast algorithm for the minimum covariance determinant estimator', *Technometrics* **41**(3), 212–223.

Rousseeuw, P. J. & Leroy, A. M. (2005), *Robust regression and outlier detection*, John wiley & sons.

Sarkar, D. (2008), 'Labels and legends', *Lattice: Multivariate Data Visualization with R* pp. 151–163.

Silge, J. & Robinson, D. (2016), 'tidytext: Text mining and analysis using tidy data principles in r', *JOSS* **1**(3).
URL: *http://dx.doi.org/10.21105/joss.00037*

Talagala, P. D., Hyndman, R. J., Smith-Miles, K., Kandanaarachchi, S. & Muñoz, M. A. (2019), 'Anomaly detection in streaming nonstationary temporal data', *Journal of Computational and Graphical Statistics* (just-accepted), 1–28.

Torgo, L. (2010), *Data Mining with R, learning with case studies*, Chapman and Hall/CRC.
URL: *http://www.dcc.fc.up.pt/ ltorgo/DataMiningWithR*

Unwin, A. (2019*a*), 'Multivariate outliers and the O3 plot', *Journal of Computational and Graphical Statistics* pp. 1–11.

Unwin, A. (2019*b*), *OutliersO3: Draws Overview of Outliers (O3) Plots*. R package version 0.6.2.
URL: *https://CRAN.R-project.org/package=OutliersO3*

Vakili, K. & Schmitt, E. (2014), 'Finding multivariate outliers with fastpcs', *Computational Statistics & Data Analysis* **69**, 54–66.

Vialaneix, N., Mariette, J., Olteanu, M., Rossi, F., Bendhaiba, L. & Bolaert, J. (2019), *SOMbrero: SOM Bound to Realize Euclidean and Relational Outputs*. R package version 1.2-4.

Wickham, H. (2007), 'Reshaping data with the reshape package', *Journal of Statistical Software* **21**(12), 1–20.
URL: *http://www.jstatsoft.org/v21/i12/*

Wickham, H. (2016), *ggplot2: Elegant Graphics for Data Analysis*, Springer-Verlag New York.
URL: *http://ggplot2.org*

Wickham, H. (2017), *tidyverse: Easily Install and Load the 'Tidyverse'*. R package version 1.2.1.
URL: *https://CRAN.R-project.org/package=tidyverse*

Wilkinson, L. (2017), 'Visualizing big data outliers through distributed aggregation', *IEEE transactions on visualization and computer graphics* **24**(1), 256–266.

Yamini, M. P. C. (2019), 'A violent crime analysis using fuzzy c-means clustering approach', *ICTACT Journal on Soft Computing* **9**(3), 1939–1944.

Zimek, A., Schubert, E. & Kriegel, H.-P. (2012), 'A survey on unsupervised outlier detection in high-dimensional numerical data', *Statistical Analysis and Data Mining: The ASA Data Science Journal* **5**(5), 363–387.