Automatic Clustering by Detecting Significant Density Dips in Multiple Dimensions

Pantelis Chronis
University of Peloponnese, Athena RC
chronis@uop.gr

Spiros Athanasiou
Athena RC
spathan@imis.athena-innovation.gr

Spiros Skiadopoulos
University of Peloponnese
spiros@uop.gr

***Abstract*—Clustering algorithms are used to find groups of similar items in a dataset. Automatic clustering algorithms achieve this task without requiring users to input critical parameters. A recent automatic clustering methodology uses Hartigan's dip test to detect significant peaks in the distribution of a dataset. This test can detect peaks in the distribution of a *one-dimensional* variable. To perform clustering in multiple dimensions, algorithms of this methodology rely on one-dimensional transformations of the dataset, which limits their effectiveness. In this paper, we present M-DIP, an automatic clustering algorithm that works directly on *multi-dimensional* space. M-DIP also assumes that clusters correspond to different peaks in the distribution of the dataset. It separates clusters at the dips that form between neighboring peaks. Dips are detected directly in multi-dimensional space, using a graph-based method. Their statistical significance is evaluated through appropriate simulations. Our experimental evaluation indicates that M-DIP achieves significantly better results than existing algorithms based on Hartigan's dip, as well as other state-of-the-art automatic clustering algorithms.**

I. INTRODUCTION

Clustering algorithms attempt to group the items of a dataset together in a meaningful way. They consider items as points in a data-space and define a cluster as a group of points that are close by, while being far from other groups of points. Most clustering algorithms depend on input parameters, like the number of clusters or thresholds for the density of a cluster. These parameters are very important and different parameter values may lead to very different results. However, in most cases there is no objective way to determine the appropriate parameters, which creates uncertainty over the optimal parametrization and the result. Automatic clustering algorithms aim to address these problems by automatically selecting parameter values, based on appropriate assumptions.

A very effective existing automatic clustering methodology assumes that clusters correspond to different peaks in the probability distribution that generated the dataset [1], [2], [3], [4]. Since the distribution is not available, these peaks must be detected from the empirical density of the dataset. However, peaks in the empirical density may be formed at random, due to sampling variance, without existing in the distribution that generated the data. To achieve accurate clustering, an algorithm needs to automatically distinguish between random peaks and peaks that actually exist in the distribution.

In *one-dimensional* space, this can be achieved using Hartigan's dip test of unimodality [5]. Given a sample from a one-dimensional random variable, this test can determine whether one or multiple peaks (modes) exist in its distribution. It achieves this by detecting the dips that are formed between neighboring peaks. Recently, notable clustering algorithms have been proposed that use Hartigan's dip to perform clustering in multidimensional space. Since the test is one-dimensional, such algorithms rely on one-dimensional transformations of the dataset. The most notable algorithms of this methodology are skinny-dip [1] and dip-means [2]. Skinny-dip applies the test on linear projections of the data and dip-means applies it on pairwise distances between points. Inevitably, part of the dataset's structure is lost after such transformations: a cluster may not be visible on any linear projection and its location may become ambiguous if we only consider pairwise point distances. Unfortunately, it is not straightforward to generalize Hartigan's dip test to multi-dimensional space.

In this paper we present M-DIP, an algorithm that can detect significant density dips *directly in multi-dimensional space*. M-DIP also assumes that clusters corresponds to different peaks in the probability distribution that generated the data, and separates clusters at the density dips that are formed between neighboring peaks. To distinguish between dips that are formed at random in the dataset and dips that actually exist in the generating distribution, M-DIP calculates bounds on the size of random dips in multi-dimensional space. These bounds are based on the simple property that random dips are not likely to become very deep. Their calculation is interdependent with the way that the dips are detected and measured. In M-DIP, the dips are detected by traversing the dataset following appropriate paths of closely located points, which pass through the denser regions of the dataset. The bounds are obtained through simulations using the uniform distribution, which, as we show in Section III-C, is an appropriate, worst case distribution for obtaining such bounds. By evaluating multi-dimensional space directly, M-DIP can detect clusters more effectively than skinny-dip, dip-means and other state-of-the-art automatic clustering algorithms

The rest of the paper is organized as follows: in Section II we describe M-DIP in detail, in Section III we present a technical analysis of M-DIP, in Section IV we survey related work in comparison with M-DIP, in Section V we present our experimental evaluation and, finally, in Section VI we preset our conclusions.

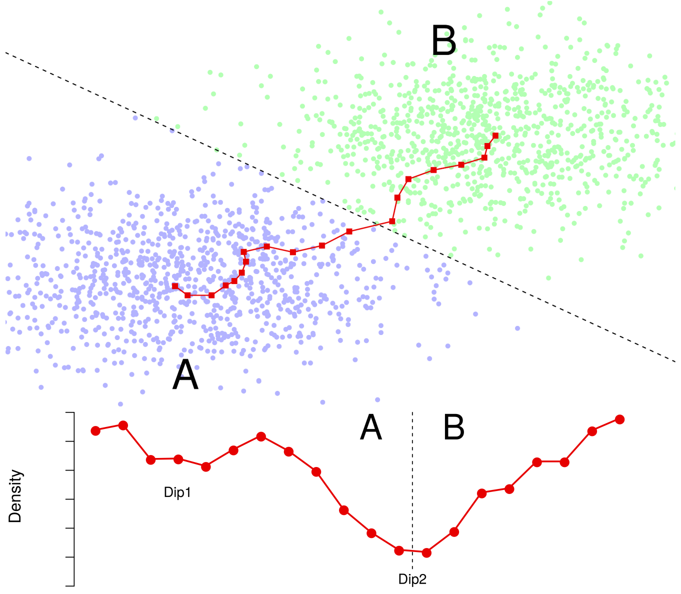


Fig. 1: An example dataset with two Gaussian clusters, A and B, and a path from A to B, that passes through closely located points. The plot at the bottom shows the density of the dataset, on each step of the path. The dashed line separates the two clusters. A significant dip (Dip2) s formed in the density at the location where the clusters are separated. A shallower dip (Dip1) also is formed at random.

II. Our method

An example for the operation of M-Dip is illustrated in Figure 1. Specifically, the figure depicts a dataset generated from a mixture of two Gaussians, A and B, and a path of closely located points that goes from A to B. The density at each step of the path is depicted at the bottom of the figure. A deep density dip (Dip2) is formed at the location where the clusters are separated. A shallower dip (Dip1) is formed randomly inside cluster A. The most important part of M-Dip is that it can differentiate between Dip1, which is random, and Dip2, which is significant, while directly traversing the multi-dimensional space. This is achieved using a threshold of statistical significance. This threshold constitutes a bound on the depth that a dip can reach at random in a dataset, without actually existing in the generating distribution. The threshold is interdependent with the way the dataset is traversed and is calculated through simulations with the uniform distribution. As we show in Section III-C, uniform is appropriate for obtaining this bound because, under certain assumptions, it generates the deepest random dips. Next we present the algorithm in detail.

A. Notation and definitions

We consider a dataset $\mathcal{X}$, consisting of n points in $\mathbb{R}^m$. Let $x \in \mathcal{X}$ be a point and x_k be its k-th nearest neighbor in $\mathcal{X}$. Let also $\rho(x, x_k)$ be the distance between x and x_k. In the hypersphere centered at x with a radius of $\rho(x, x_k)$ there are $k+1$ points of $\mathcal{X}$ (x and its k-neighbors). In other words, around x and in a volume proportional to $\rho(x, x_k)^m$ there are

Notation	Explanation
$\mathcal{X}$	A set of n points in $\mathbb{R}^m$.
$dens(x)$	The density of point x.
k-NN$(\mathcal{X})$	The k nearest neighbor graph of dataset $\mathcal{X}$.
$\rho(x, y)$	The distance between points x and y.
p_{xy}	A path connecting two points x and y of k-NN$(\mathcal{X})$
$dip(p)$	The dip, i.e., the minimum density of path p.
$\mathcal{P}_{xy}$	The set containing all path that connect two points x and y of k-NN$(\mathcal{X})$
$dip_s(\mathcal{P})$	The shallowest dip in the set of paths $\mathcal{P}$.
$point_s(\mathcal{P})$	The shallowest dip point of $\mathcal{P}$.
$rsize_s(\mathcal{P})$	The relative size of the shallowest dip.

TABLE I: Notation summary

$k+1$ points. We define the *density* of x, denoted by $dens(x)$, as:

$$dens(x) = \frac{k+1}{\rho(x, x_k)^m} \tag{1}$$

The *k-neighbor graph* of dataset $\mathcal{X}$, denoted by k-NN$(\mathcal{X})$, is a directed graph with nodes the points of $\mathcal{X}$ and directional edges that point from every node towards the nodes of its k-nearest neighbors. Intuitively, k-NN$(\mathcal{X})$ connects all points of $\mathcal{X}$ using paths that exclusively contain k-nearest neighbors. We denote as ψ the *most dense* point of $\mathcal{X}$. Let us now consider a path $p_{\psi x}$ that connects the most dense point ψ with an arbitrary point x in k-NN$(\mathcal{X})$. The *dip* of $p_{\psi x}$, denoted by $dip(p_{\psi x})$ is the minimum density of the points in $p_{\psi x}$. More formally, we have $dip(p_{\psi x}) = \min\{dens(v) \mid v \in p_{\psi x}\}$. There are two cases regarding $dip(p_{\psi x})$:

- If $dip(p_{\psi x})$ occurs at the ending point x, then density follows a decreasing trend along path $p_{\psi x}$.
- If $dip(p_{\psi x})$ does not occur at the endpoint x, then a *density dip* is formed along p. This means the density starts at a certain high level on point ψ, drops to the lowest level (denoted by $dip(p_{\psi x})$) and then increases again at the ending point x.

Let $\mathcal{P}_{\psi x}$ be the set of *all* paths that connect most dense point ψ with a point x in k-NN$(\mathcal{X})$. The *shallowest dip* of set $\mathcal{P}_{\psi x}$, denoted by $dip_s(\mathcal{P}_{\psi x})$, is defined as the shallowest dip of all its paths, i.e., the dip that has the maximum density. More formally, we have $dip_s(\mathcal{P}_{\psi x}) = \max\{dip(p) \mid p \in \mathcal{P}_{\psi x}\}$. Intuitively, $dip_s(\mathcal{P}_{\psi x})$ is the most shallow dip that needs to be traversed to get from ψ to x. We also define the *shallowest dip point* of set $\mathcal{P}_{\psi x}$, denoted by $point_s(\mathcal{P}_{\psi x})$, as the point of the path in $\mathcal{P}_{\psi x}$ that achieves the shallowest dip. More formally, we have $point_s(\mathcal{P}_{\psi x}) = \{v \in p,\ p \in \mathcal{P}_{\psi x} \mid dens(v) = dip(p) = dip_s(\mathcal{P}_{\psi x})\}$.

Finally, we define the *relative size of the shallowest dip* of $\mathcal{P}_{\psi x}$, denoted by $rsize_s(\mathcal{P}_{\psi x})$, as

$$rsize_s(\mathcal{P}_{\psi x}) = \frac{dip_s(\mathcal{P}_{\psi x})}{dens(x)} \tag{2}$$

$rsize_s(\mathcal{P}_{\psi x})$ is the metric that M-Dip uses to evaluate and distinguish dips. The value of $rsize_s(\mathcal{P}_{\psi x})$ represents the density dip than needs to be traversed in k-NN$(\mathcal{X})$, in order to get from ψ to x, normalized with the density $dens(x)$ of the common ending point x of the paths in $\mathcal{P}_{\psi x}$. The way that

$rsize_s(\mathcal{P}_{\psi x})$ is defined means that on every path from ψ to x the relative density (with respect to $dens(x)$) falls at least as low as $rsize_s(\mathcal{P}_{\psi x})$, i.e., there is no way to get from ψ to x without traversing a dip with relative density at least as deep as $rsize_s(\mathcal{P}_{\psi x})$.

B. Threshold of statistical significance

We assume data are generated from an underlying distribution $\mathcal{D}$. Significant dips, that exist in distribution $\mathcal{D}$, are distinguished from random dips, that are formed randomly on dataset $\mathcal{X}$, by their depth. Dips that occur at random are not likely to exceed a certain depth. This is quantified by threshold θ. Comparing $rsize_s(\mathcal{P}_{\psi x})$ with θ takes the form of statistical hypothesis testing. The null hypothesis H_0 is that the distribution $\mathcal{D}$ does not contain a dip (i.e., it contains only a single peak, it is unimodal). The alternative hypothesis H_1 is that the distribution contains a dip (i.e., it contains more than one peaks, it is multimodal). We want a threshold θ such that, under hypothesis H_0, a point x with $rsize_s(\mathcal{P}_{\psi x}) < \theta$ exist in $\mathcal{X}$ with probability smaller than α, where α is the significance level of the test. If we observe $rsize_s(\mathcal{P}_{\psi x}) < \theta$ for any x, we can reject H_0 in favour of H_1, at significance level α. Therefore, if $rsize_s(\mathcal{P}_{\psi x}) < \theta$ we infer that the distribution has multiple peaks.

Due to the complexity of $rsize_s(\mathcal{P}_{\psi x})$ and the multiple and dependent comparisons of $rsize_s(\mathcal{P}_{\psi x})$ with θ for all x, the analytical calculation of θ is intractable. In order to obtain a value for θ we resort to simulations. During the simulations we estimate how low $rsize_s(\mathcal{P}_{\psi x})$ can become at random, without the existence of multiple peaks. As we show in Section III-C, under certain assumptions, the appropriate distribution for these simulations is the uniform, because it generates the minimum values for $rsize_s$ at random, among all unimodal distributions.

The simulations are performed in the following way. We generate r random datasets $\mathcal{X}_1, \ldots, \mathcal{X}_r$ each containing n points (i.e, the number of points that $\mathcal{X}$ has). These points are randomly drawn from the *m-dimensional uniform distribution* over the region $[0,1]^m$. For each dataset $\mathcal{X}_i$, $1 \le i \le r$:

- We construct k-NN($\mathcal{X}_i$).
- We compute the density ($dens$) of all points in $\mathcal{X}_i$.
- We select the point ψ having maximum density in $\mathcal{X}_i$ and compute the smallest dip $rsize_s(\mathcal{P}_{\psi x})$ for all points in $x \in \mathcal{X}_i$.
- We return the smallest dip observed in $\mathcal{X}_i$, denoted by $min.rsize_s(\mathcal{X}_i)$.

Then, threshold θ is assigned the $1-\alpha$-percentile among the $min.rsize_s(\mathcal{X}_i)$ values.

Calculated this way, threshold θ provides a bound for the value of $rsize_s$ on a dataset generated from the uniform distribution. Since the uniform is the worst case unimodal distribution, θ is a bound for other unimodal distributions as well. We note that threshold θ implicitly corrects for the multiple dependent comparisons of $rsize_s(\mathcal{P}_{\psi x})$ with θ for all x, because it is calculated as a bound for the *minimum* $rsize_s(\mathcal{P}_{\psi x})$ for all x in the entire dataset, and not for each point independently. Also, we note that, by definition of $rsize_s(\mathcal{P}_{\psi x}$, condition $rsize_s(\mathcal{P}_{\psi x}) < \theta$ explicitly evaluates the density decrease relative to $dens(x)$. However, implicitly, it also tests the density decrease relative to ψ, since ψ is selected as the most dense point of the dataset and $dens(\psi) > dens(x)$. Finally, we note that threshold θ is used to infer the existence of a dip in the distribution. Having inferred its existence, M-DIP uses the location of the shallowest dip ($point_s(\mathcal{P}_{\psi x})$) to divide the points between the two sides of the dip.

The exact way the algorithm traverses the dataset to obtain $rsize_s$ and apply θ to detect clusters is described next.

```
Algorithm: SHALLOWDIPSEARCH
Input      : A point dataset $\mathcal{X}$
Parameter  : Statistical significance threshold $\theta$
Output     : Cluster $\mathcal{A}$
1  $\mathcal{A} = \emptyset$, $\mathcal{B} = \emptyset$ , $\mathcal{C} = \mathcal{X}$
2  Let $\psi$ be the most dense point of $\mathcal{X}$
3  Move $\psi$ from set $\mathcal{C}$ to set $\mathcal{A}$
4  Set $cur$ to be $\psi$
5  repeat
6      Select points $N_{cur}$ that are neighbors of $cur$ in
         $k$-NN($\mathcal{X}$)
7      forall $x$ in $N_{cur}$ do
8          if $x$ in $C$ then
9              Set $rsize_s(\mathcal{P}_{\psi x}) =$
                 $min(rsize_s(\mathcal{P}_{\psi cur}), dens(cur))$
10             Move $x$ from set $\mathcal{C}$ to set $\mathcal{B}$
11     Let $\beta$ be the most dense point of $\mathcal{B}$
12     Remove $\beta$ from $\mathcal{B}$
13     if $\theta < rsize_s(\mathcal{P}_{\psi\beta})$ then
14         Add $\beta$ into $\mathcal{A}$
15         Set $cur$ to be $\beta$
16     else
17         Remove from $\mathcal{A}$ points visited following neighbors
             of $point_s(\mathcal{P}_{\psi\beta})$
18 until $\mathcal{B} = \emptyset$;
19 return $\mathcal{A}$
```

Fig. 2: Smallest Dip Search

C. *Algorithm* SHALLOWDIPSEARCH

Algorithm SHALLOWDIPSEARCH (Figure 2) takes as input a dataset $\mathcal{X}$ and uses threshold θ to return a cluster $\mathcal{A}$. The algorithm maintains three sets of points $\mathcal{A}$, $\mathcal{B}$ and $\mathcal{C}$. Set $\mathcal{A}$ contains the visited points which will form the returned cluster, set $\mathcal{B}$ contains the neighbors of points in $\mathcal{A}$ and set $\mathcal{C}$ contains all other points. Initially, $\mathcal{A} = \emptyset$, $\mathcal{B} = \emptyset$ and $\mathcal{C} = \mathcal{X}$ (Line 1).

SHALLOWDIPSEARCH computes the most dense point ψ of $\mathcal{X}$ (Line 2), moves it from set $\mathcal{C}$ to set $\mathcal{A}$ (Line 3) and sets it as the current processing point (stored in variable cur – Line 4). Then, SHALLOWDIPSEARCH performs a repeat loop (Lines 5–18) until set $\mathcal{B}$ is empty. In each iteration, the algorithm *(a)* finds the k-nearest neighbors of cur and if they are in $\mathcal{C}$, calculates their $rsize_s$ (Line 9) and moves them to $\mathcal{B}$ (Line 10), *(b)* finds the most dense point β of $\mathcal{B}$ (Line 11) and removes it from $\mathcal{B}$ (Line 12) and *(c)* if $\theta < rsize_s(\mathcal{P}_{\psi\beta})$ then β belongs to the same cluster as ψ, so it is added to $\mathcal{A}$

```
Algorithm: M-Dip
Input   : A point dataset $\mathcal{X}$
Output  : A set of clusters $\mathcal{S}$
1 $\mathcal{S} = \emptyset$
2 repeat
3     Calculate statistical significance threshold $\theta$
4     $\mathcal{A} = $ ShallowDipSearch$(\mathcal{X}, \theta)$
5     Add the new cluster $\mathcal{A}$ into the set of clusters $\mathcal{S}$
6     Remove the points of $\mathcal{A}$ from $\mathcal{X}$
7 until $\mathcal{X}$ is empty;
8 return $\mathcal{S}$
```

Fig. 3: Significant Density Peak Detection

(Line 14) and is set as the current processing point cur (Line 15). If $rsize_s(\mathcal{P}_{\psi\beta}) \leq \theta$, the dip towards β is considered deep and β is not included in cluster $\mathcal{A}$. In this case the two clusters are separated on $point_s(\mathcal{P}_{\psi\beta})$, where the dip of the path that connected ψ to β lies. Points that have been included in $\mathcal{A}$ following paths through $point_s(\mathcal{P}_{\psi\beta})$ lie past the detected dip, so they are considered part of the neighboring cluster and are removed from $\mathcal{A}$ (Line 17). As we show in Section III-A, ShallowDipSearch correctly traverses the smallest dip paths of $\mathcal{X}$.

D. *Algorithm* M-Dip

Algorithm M-Dip (illustrated in Figure 3) iteratively applies ShallowDipSearch, to detect the clusters of a dataset. On each iteration, it first calculates the threshold of statistical significance θ (Line 3). Subsequently, it applies ShallowDipSearch, with the calculated threshold, to detect a single cluster $\mathcal{A}$ (Line 5). The cluster is appended to the set of discovered clusters $\mathcal{S}$ and its points are removed from $\mathcal{X}$ (Line 5). The process is iterated until all points of $\mathcal{X}$ are assigned to a cluster. Threshold θ must be updated in every iteration of M-Dip because it depends on the number of points remaining in $\mathcal{X}$, after the removal of cluster $\mathcal{A}$.

III. Analysis of M-Dip

A. *Shallow dip search*

In this subsection we show that ShallowDipSearch traverses the shallowest dip path between point ψ and each point x it visits. We note that the algorithm stops traversing points on paths where condition $\theta < rsize_s(\mathcal{P}_{\psi x})$ is violated. These points by definition belong to a different cluster and will be traversed during a next iteration. We focus on the algorithm's behaviour on points that are traverse on a given iteration. Also, we note that it is possible for the k-NN graph to not be fully connected. In this case the algorithm will be unable to reach points in different components of the graph. This can be easily overcome by adding one auxiliary point and two auxiliary edges, to connect any two disconnected components. The way these points are used is trivial so we do not describe it in detail.

The neighbors of x in k-NN($\mathcal{X}$) are its k nearest neighbors, denoted as $\mathcal{NN}(x)$. An edge from x to y exists in k-NN($\mathcal{X}$), for each $y \in \mathcal{NN}(x)$.

Lemma 1. *In k-NN($\mathcal{X}$), a path that contains point x also contains one of its k nearest neighbors $\mathcal{NN}(x)$, unless x is the final point of the path.*

Lemma 1 follows directly from the definition of k-NN($\mathcal{X}$).

Lemma 2. *At each iteration of* ShallowDipSearch*, any path from ψ to any point $x \in \mathcal{C}$ passes through a point in $\mathcal{B}$.*

Proof. Let $\mathcal{B}_i$ be the set $\mathcal{B}$ at iteration i. At iteration 1, B_1 contains $\mathcal{NN}(\psi)$, the neighbors of ψ. By Lemma 1, all paths from ψ to $x \in \mathcal{C}$ ($x \neq \psi$) pass from a point in $\mathcal{B}_1$. Therefore Lemma 2 holds for $i = 1$. At iteration i the algorithm removes β from $\mathcal{B}$ and adds $\mathcal{NN}(\beta)$ (ShallowDipSearch Line 12): $\mathcal{B}_i = (\mathcal{B}_{i-1} \setminus \beta) \cup \mathcal{NN}(\beta)$. By Lemma 1, every path from ψ to $x \in \mathcal{C}$ that passes through β ($\beta \neq x$), also passes through a point in $\mathcal{NN}(\beta)$. Consequently, by removing β and adding $\mathcal{NN}(\beta)$ to $\mathcal{B}$, no paths are excluded. Therefore, if Lemma 2 holds for $\mathcal{B}_{i-1}$ it also holds for $\mathcal{B}_i$. By induction this proves Lemma 2. □

Proposition 1. *Algorithm* ShallowDipSearch *traverses the shallowest dip path from point ψ to every other point x*

Proof. Let u denote the shallowest dip point on the shallowest dip path p from ψ to x discovered by ShallowDipSearch. We assume that another path p' from ψ to y exists, with shallowest dip point u' having:

$$dens(u) < dens(u') \tag{3}$$

This would mean that the path discovered by ShallowDipSearch does not contain the shallowest dip, since the density of u' is higher than u. By definition of the minimum density point, each point in path p' has higher density than u':

$$dens(u') \leq dens(v), \ \forall v \in p' \tag{4}$$

Let i be the iteration of ShallowDipSearch on which u was selected and included in $\mathcal{A}$. We denote as $\mathcal{B}_i$ the set $\mathcal{B}$, on iteration i, right before the selection of u (ShallowDipSearch Line 11). By Lemma 2, a point v from the alternative path p' exists in $\mathcal{B}_i$. For $dens(u)$, $dens(u')$ it holds from ShallowDipSearch Line 11 and Equation 4:

$$\begin{aligned} dens(z) < dens(u), \forall z \in B_i \Rightarrow dens(v) < dens(u) \\ \Rightarrow dens(u') < dens(u) \end{aligned} \tag{5}$$

This contradicts Equation 3 that holds by definition of p' as a path with a shallower dip than p. Therefore p' can not exist and p, traversed by ShallowDipSearch, is the shallowest dip path from ψ to x. □

B. *Estimation of $dens(x)$*

Density $dens$ is calculated using the k-th nearest neighbor of x. This method of density estimation has the advantage that it does not require the selection of a bandwidth parameter, which is difficult to determine and may vary through a dataset. Instead, it adjusts automatically to the density of each location. However this method requires the selection of parameter k.

We assume that dataset $\mathcal{X}$ consists of points sampled randomly from the underlying probability distribution $\mathcal{D}$. This makes $dens(x)$, the empirical density of point x, a random variable, affected by sampling variance. Parameter k affects the variance of $dens(x)$, higher k leads to smaller variance. The value of k also determines the resolution with which the density over the data-space is calculated. In this case a higher k leads to smaller resolution. We want to find a value for k that achieves low variance while maintaining high resolution. To find such a value we calculate the variance of $dens(x)$. (This analysis of $dens(x)$ is also useful for the analysis of the uniform distribution, in Section III-C.)

Proposition 2. *The probability distribution of $dens(x)$ is $p(dens(x) < d) = F(k-1, n-1, \pi_x(e))$, where F is the Cumulative Distribution Function (CDF) of the Binomial distribution, $\pi_x(e)$ is the total probability mass of dataset's distribution $\mathcal{D}$, in a hypersphere of radius $e = \sqrt[m]{\frac{k+1}{d}}$ around point x.*

Proof. $dens(x)$ is defined in Equation 1. The probability that $\rho(x, x_k)$ (the distance between x and its k nearest neighbor x_k) is smaller than a constant e is equal to the probability that k or more points fall in distance e from x. This can be modelled using the binomial distribution as:

$$p(\rho(x, x_k) \leq e) = \sum_{j=k}^{n-1} \binom{n-1}{j} \pi_x(e)^j (1-\pi_x(e))^{n-1-j} \tag{6}$$

For a given j the above expression corresponds to the probability that j or more out of the $n-1$ remaining points of $\mathcal{X}$ fall in radius e from x. $\pi_x(e)$ equals to the total probability mass in an area of radius e around x. For any underlying distribution $\mathcal{D}$, $\pi_x(e)$ is a constant in $[0, 1]$, which depends on $\mathcal{D}$, the location of x and radius e.

The Binomial CDF F gives the probability of an event to occur up to k times with n trials, therefore its complementary $1-F$ gives the probability for more than k times:

$$1 - F(k, n, \pi) = \sum_{i=k+1}^{n} \binom{n}{i} \pi^i (1-\pi)^{n-i} \tag{7}$$

From Equations 6 and 7:

$$p(\rho(x, x_k) \leq e) = 1 - F(k-1, n-1, \pi_x(e)) \tag{8}$$

Consequently, because dens is inverse proportional to ρ^m (Equation 1):

$$\begin{aligned} p(dens(x) < d) &= 1 - p(\rho(x, x_k) \leq e) \\ &= F(k-1, n-1, \pi_x(e)) \ , \quad e = \sqrt[m]{\frac{k+1}{d}} \end{aligned} \tag{9}$$

□

Using Proposition 2, we can calculate the distribution of $\pi_x(e)$ for given k and n. This distribution is approximately normal, as expected from a sum of binomials that have large n. If we approximate distribution $\mathcal{D}$ around point x using a constant z, using the distribution of $\pi_x(e)$, we can estimate

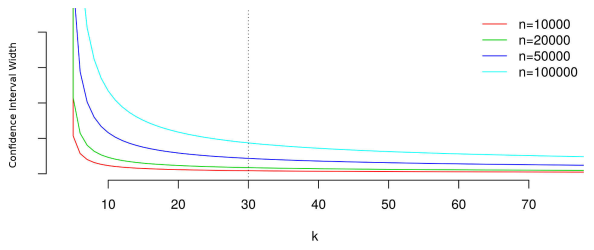


Fig. 4: The width of the 95% confidence interval of $dens(x)$, relative to number of neighbors k, for various dataset sizes n. The scale of the y axis depends on probability density z around x

the distribution of $dens(x)$, as a function of n, k, z. This approximation is good when n is large or $\mathcal{D}$ is smooth. Then $\pi_x(e) = z \cdot V(e)$, $V(e) = \frac{\pi^{\frac{m}{2}}}{\Gamma(\frac{m}{2}+1)} \cdot e^m$, where $V(e)$ is the volume of the m-dimensional hypersphere with radius e. Since z and $\frac{\pi^{\frac{m}{2}}}{\Gamma(\frac{m}{2}+1)}$ are constants we denote them as $c_{z,m}$:

$$\pi_x(e) = c_{z,m} \cdot e^m \Leftrightarrow e^m = \frac{1}{c_{z,m}} \pi_x \tag{10}$$

Using the distribution of $\pi_x(e)$ and Equation 10, we calculate the distribution of $\rho(x, x_k)^m$ and using Equation 1 we calculate the distribution of $dens(x)$ and obtain the width of its 95% confidence interval as a function of k. These calculations are performed numerically, for various values of n, k. The results are depicted in Figure 4. The width of the confidence interval decreases rapidly for small k and then it stabilizes. Moreover, the width also increases for larger n. Probability density z of the dataset is a scale parameter, which affects the scale of the y-axis but not the shape of the curves. From Figure 4 we can see that the estimation of density has converged for most dataset sizes n. Therefore, as a default value we select $k = 30$. For very large datasets a larger value of k may be selected. When n increases, the error of the estimation increases but the mean of the estimation increases as well. If we are interested in the relative error of the estimation the $k = 30$ is appropriate for all dataset sizes. Finally, the above numerical analysis suggests, that for $k = 30$, the distribution of $dens(x)$ is approximately normal with standard deviation to mean ratio approximately 0.18, for all n (given $n \geq k$).

C. *Uniform distribution*

To detect clusters, M-DIP evaluates the *unimodality* of the distribution that generated the dataset. We consider a distribution unimodal if it has a single maximum which is global, i.e., it has no local maxima. Although other definitions for multivariate unimodality exist, this simple definition is sufficient for our application. An example of a multivariate unimodal distribution is depicted in Figure 5. Unimodal distributions with more complex shapes exist, as long as no additional local maxima are formed. By definition, a unimodal distribution has no dips that *completely* separate any two locations of the space. Therefore any dips that completely separate two locations in a dataset generated from a unimodal distribution are *random*. The size of such a random dip is

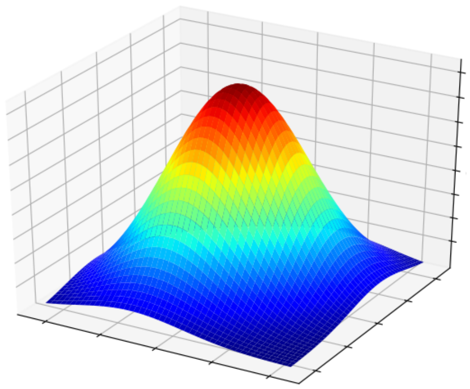


Fig. 5: An example of a two dimensional unimodal distribution

unlikely to become very large. We want to find the unimodal distribution that generates the *largest* random dips, in order to obtain bounds for their size.

In a dataset $\mathcal{X}$, M-DIP evaluates the size of a dip using metric $rsize_s(\mathcal{P}_{\psi x})$. Metric $rsize_s(\mathcal{P}_{\psi x})$ expresses the density drop on *all* paths from peak ψ to x. Its behaviour can be studied using the idea of a vertex-cut set. On a graph G a vertex-cut set is a set of nodes which, if removed, G is no longer connected. We say that a vertex-cut set C separates x and ψ if, when C is removed, x is not reachable from ψ. We note that, in k-NN graph, nodes correspond to points of the dataset. An example of a vertex cut set that separates ψ and x is depicted in Figure 6. If all nodes in the box denoted by C are removed, x is not reachable from ψ. The background color represents the probability density on each location of the data-space (red is denser). Given a point x, a vertex-cut set can contain nodes in regions with higher and lower probability density than x. Generally, more nodes of the set are expected to exist in regions with higher probability density.

From the definition of $rsize_s(\psi, x)$ (Section II-A), in order for $rsize_s(\psi, x) < \theta$, a vertex-cut set C which separates ψ from x must exist for which $dens(u) < \theta \cdot dens(x), \forall u \in C$. In this case a density drop of relative size θ is formed all around point x. Next, we present an analysis that shows that under some assumptions, the probability of $rsize_s(\psi, x) < \theta$ decreases when the distribution becomes non uniform uniform around x. These assumptions are very simplifying but can be generalised.

In brief, the probability that $dens(u) < \theta \cdot dens(x), \forall u \in C$ (i.e., the probability falls below a specified level for *all* points of a cut-set C)is expressed as a product. Increasing the density of some regions and decreasing the density of other regions around x increases some factors of the product and decreases others. Since a product is more sensitive to small factors, the overall probability decreases. This is magnified by the fact that more points fall in the denser regions, therefore there are more decreasing factors. Next we analytically describe this effect under the assumption of a linearly approximated distribution, in a location close to x.

More specifically, we consider point x in the center of a region R. We assume that probability density is linear in region

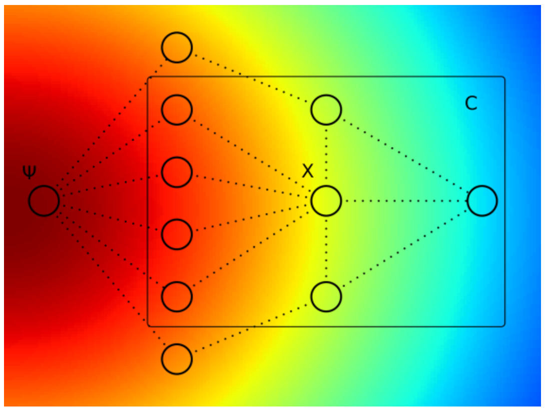


Fig. 6: An illustration of a vertex-cut set that sparates ψ from x. All paths from ψ to x pass from a node in C. The background color represents the probabillity density of the space. More nodes exist in the denser part of C.

R. An illustration of this setting is depicted in Figure 7. Given that the probability density is linear in R and x is at the center of R, the probability density is symmetric around line L (or hyperplane L in more dimensions), which passes from x and is perpendicular to the direction of the gradient of the distribution. We divide region R in two sub-regions R_1 and R_2 divided by line L. Since L is perpendicular to the gradient, R_1 has higher probability density than R_2. We further divide R_1 and R_2 into many small subregions as depicted by the boxes in Figure 7 (the specific shape of the subregions is not important). Given the symmetry around L for each subregion r in R_1, a antisymmetric subregion $r_{(s)}$ exists in R_2.

We denote as π_r the total probability mass in the box of subregion r. From the antisymmetry it holds that $\pi_r = \pi_x + c$ and $\pi_{r_{(s)}} = \pi_x - c$, for constant $c \geq 0$. We assume that the number of nodes in cut-set C, $|C| = l$, is independent of the distribution in R and that the nodes are randomly and independently distributed on region R. From the binomial distribution, on average, the number l_r of nodes on each subregion r will be proportional to the probability mass of the region: $l_r = \beta l \pi_r$,

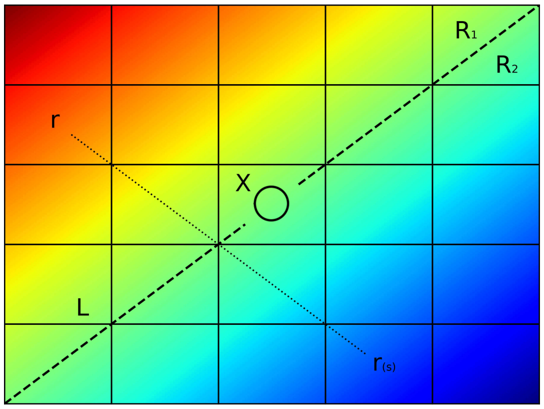


Fig. 7: A point x in a region with linear probability distribution. The region in divided in many subregions. The density of the subregions is antisymmetric to line L (e.g., subregions r, $r_{(s)}$ are antisymmetric).

where β is the total density mass of R: $\beta = \sum \pi_r, \forall r \in R$. We denote as $p_{dip}(u|\pi_u) = p(dens(u) < \theta \cdot dens(x)|\pi_u, \pi_x)$ the probability that the empirical density at point u ($dens(u)$) falls bellow $\theta \cdot dens(x)$. Probability $p_{dip}(u|\pi_u)$ depends on the probability densities π_u, π_x around each point respectively, but we omit π_x because it is constant for all u. We denote as $p_{dip}(u_r|\pi_r)$ the same probability for a single node $u_r \in r$. It is important to remember that as the probability density increases the empirical density increases as well (Section III-B). Therefore $p_{dip}(u_r|\pi_r)$ is decreasing with π_r.

From the description presented above, and assuming the events are independent, the probability that a significant density decrease occurs on all points of C is:

$$\begin{aligned} p(C) &= \prod_{u \in C} p_{dip}(u|\pi_u) \\ &= \prod_{r_1 \in R_1} p_{dip}(u_{r_1}|\pi_{r_1},)^{\beta l \pi_{r_1}} \prod_{r_2 \in R_2} p_{dip}(u_{r_2}|\pi_{r_2})^{\beta l \pi_{r_2}} \\ &= \prod_{r \in R_1} p_{dip}(u_r|\pi_x + c)^{\beta l(\pi_x + c)} p_{dip}(u_{r(s)}|\pi_x - c)^{\beta l(\pi_x - c)} \end{aligned} \tag{11}$$

In the final product of Equation 11 we have paired each subregion $r \in R_1$ with its symmetric $r_{(s)} \in R_2$. Each pair consists of term $p_{dip}(u_r|\pi_x + c)$, which has lower probability, and $p_{dip}(u_r|\pi_x - c)$, which has higher probability, because $c > 0$. Moreover, the terms with lower probability have larger exponents: $\beta l(\pi_x + c) > \beta l(\pi_x - c)$. The product of Equation 11 decreases very fast when c grows, under any reasonable approximation for $p_{dip}(u_r|\pi_x + c)$. This occurs because the product is more sensitive to terms approaching zero than to terms approaching one (smaller factors have larger partial derivatives). The effect is magnified by the exponents, which are larger for the smaller factors. If we follow the analysis of Section III-B and approximate $p(dens(x)|\pi_x)$ with a Normal distribution with mean and standard deviation proportionate to π_x, we can calculate $p_{dip}(u_r|\pi_x + c)$ using the Normal CDF. Then we can prove that the probability of Equation 11 is maximized for $c = 0$, which corresponds to the uniform distribution. Due to factor l in the exponents of Equation 11, the probability of a dip in very large sets C, that span regions far from x, approaches zero.

This analysis shows that the uniform is the worst case distribution for the cases that are *well approximated by our assumptions*. Specifically, our assumptions are: (a) we consider a location close to x when the distribution is linearly approximated, (b) the size of cut set C is independent of the distribution and (c) the nodes of cut set C are distributed multinationally in the subregions of R. These assumptions are restrictive but they serve to show that the probability that a dip occurs all around x diminishes very quickly when a non-zero gradient in density exists at the region around x. This still holds even when assumptions (a),(b) and (c) are significantly relaxed. We do not have a proof showing how general these assumptions can become. However, we expect that they can be relaxed enough to cover all well known unimodal distributions that are decreasing smoothly around their modes. Spurious distributions, that can not be well approximated by similar assumptions, might also exist. Encountering a dataset with such a distribution in practice will lead to an increased false positive cluster discovery rate. However, we consider such distributions to be uncommon in real world datasets. Overall, since we are interested in the practical application of clustering, based on the analysis presented in this section, we select the uniform as the appropriate worst case distribution for M-DIP.

Proposition 3. *On a dataset $\mathcal{X}$ generated from distribution $\mathcal{D}$, the distribution of $rsize_s(\mathcal{P}_{\psi x})$ for any $x \in \mathcal{X}$ is invariant to the scale of $\mathcal{D}$.*

Proof. Each element $x^{(A)} \in \mathcal{X}^{(A)}$ is an m-dimensional vector with coordinates $x_j^{(A)}, 1 \leq j \leq m$. We the scaled version $X^{(B)}$ with $x_j^{(B)} = c \cdot x_j^{(A)}$, for constant c. The euclidean distance between the points of each dataset is also scaled: $\|x^{(B)} - u^{(B)}\| = \|cx^{(A)} - cu^{(A)}\| = c\|x^{(A)} - u^{(A)}\|$

Since all distances are scaled, the ordering of distances does not change. Therefore the k-nearest neighbors of each point and the k-NN graph remain the same. The distance between each point and its k-th nearest neighbor is scaled, therefore the density is scaled inversely: $dens(x^{(B)}) = \frac{dens(x^{(A)})}{c^m}$

Since the k-NN graph does not change the minimum dip point remains the same for all x, with scaled density $dip_s(\mathcal{P}_{\psi,x}^{(A)}) = \frac{dip_s(\mathcal{P}_{\psi,x}^{(B)})}{c^m}$. From Equation 2 $rsize_s$ is calculated as a ratio of density, which are both scaled by the same constant, therefore:

$$rsize_s(\mathcal{P}_{\psi,x}^{(B)}) = \frac{\frac{dip_s(\mathcal{P}_{\psi,x}^{(A)})}{c^m}}{\frac{dens(x^{(A)})}{c^m}} = rsize_s(\mathcal{P}_{\psi,x}^{(A)}) \tag{12}$$

Since datasets generated from scaled distributions have a 1-1 correspondence and $rsize_s$ is equal for the corresponding datasets, $rsize_s$ is invariant to changes in the scale of the distribution. □

Proposition 3 allows us to use the uniform distribution in the interval $[0,1]^m$, irrespective of the density of a dataset.

D. Computational Complexity

M-DIP performs c iterations, one for each cluster of the dataset. For each iteration, it performs r iterations to calculate threshold θ. On each such iteration it generates a random dataset builds the k-NN graph, and traverses it using SHALLOWDIPSEARCH. SHALLOWDIPSEARCH is equivalent to Dijkstra's algorithm and has complexity $O(n \cdot \log n + k \cdot n) = O(n \cdot \log n)$ on the k-NN graph, that has $k \cdot n$ edges. Constructing the k-NN graph is the most demanding part of M-DIP. Using a brute force algorithm, which explicitly calculates all pairwise distances, the k-NN graph is built in $O(n^2)$. Therefore, the total complexity of M-DIP is $O(rcn^2)$.

This complexity can be significantly improved. From Proposition 3 we know that the threshold is independent from the scale of the dataset, therefore it is possible to be calculate threshold once, off-line, for various values of n, and interpolate them to obtain a curve with the appropriate θ for each n. this would eliminate the need for the simulations to be repeated each time and would make complexity $O(cn^2)$. The dependency on the number of clusters remains because after removing the points of a cluster the k-NN graph of the dataset must be modified, which we assume in the worst case is equivalent to recreating it. Finally, we note that approximate algorithms exist that construct the k-NN graph with complexity as low as $O(n^{1.14})$ [6], which can scale to large datasets.

IV. Related work

The algorithms most closely related to M-Dip are skinny-dip [1] and dip-means [2]. Both algorithms rely on Hartigan's dip test [5]. This test can detect multiple peaks in the distribution of a one-dimensional random variable. It also uses the uniform as a worst case distribution, to calculate the level of statistical significance. However, the statistic used to measure the dips is based on the empirical CDF of the sample and there is no straightforward way to generalise it to multiple dimensions, especially considering clusters of arbitrary shape.

Skinny-dip [1] applies the test on one-dimensional linear projections of the dataset. The algorithm's assumption is that clusters will appear as distinct density peaks in the distribution of a projection. Many projections are evaluated iteratively and, if multiple peaks are detected, the dataset is partitioned based on the specific projection. The partitioning is done recursively so that clusters can be separated in more than one dimensions in the final result. Skinny-dip is also able to handle background noise (i.e., points that do not belond to a cluster) which, however, can only be of uniform distribution. Skinny-dip is shown to outperform many automatic clustering algorithms, such as ric [7], sync [8] and stsc [9].

Dip-means [2] aplies the test on pairwise distances between points of a dataset. For each point the distances to all other points are calculated. The algorithm assumes that multiple clusters will appear as distinct peaks in the distribution of distances of some points with all others. These points are called split-viewers. In this case as well, the dip test is used to detect multiple peaks. A cluster is assumed to exist if more than a percentage of points act as split viewers (1% by default). After detecting the existence of multiple peaks, dip-means partitions the points using k-means. Dip-means is shown to outperform other well known algorithms based on k-means such as x-means [10] g-means [11] and pg-means [12].

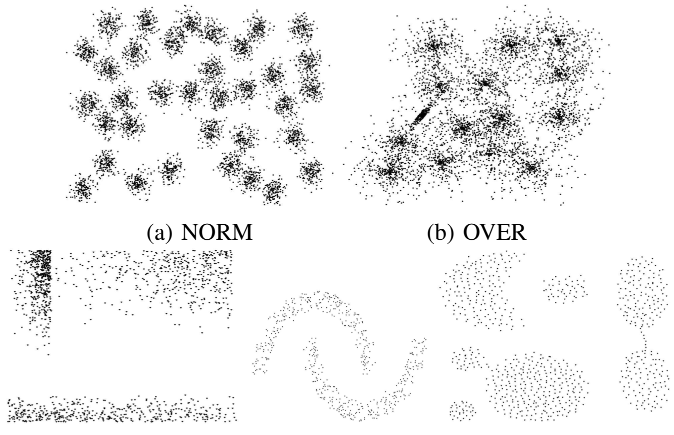
As already mentioned in Section I, the limitation of both algorithms is their reliance on one dimensional transformations. Clusters that are separable in multiple dimensions may be overlapping on each of the applied one-dimensional transformations and vice versa. Moreover, iteratively applying the test on multiple transformations changes the test's properties, due to multiple comparisons [13].

Aside from the density dip methodology, the defining characteristic of M-Dip is that it is automatic. In the following we present other notable automatic clustering algorithms. In density based clustering, the most notable automatic algorithm is hdbscan [4]. Hdbscan partitions the dataset using multiple thresholds for density, which it obtains using a heuristic called excess of mass. Its result is equivalent to running dbscan for all possible density thresholds and selecting those that are considered important based on the excess of mass metric. Hdbscan performs this with computational efficiency. Another class of density based algorithms relies on [14], which detects clusters by considering, for each point in a dataset, its density and the distance to the closest point with higher density. [14] requires the number of clusters as input. The algorithm presented in [15] approximates the distribution of the density-distance related metric using a heavy tailed distribution and uses this approximaiton to select the number of clusters. The algorithm presented in [3], selects the number of clusters using the well known silhouette [16] index.

Several automatic algorithms exist in the family of k-means, aside from dip-means. Gap [17] calculates thresholds of statistical significance for within-cluster dispersion for a clustering result. These thresholds are calculated through simulations with the uniform distribution, and are used to select a value for k. Other algorithms include x-means [10], which recursively partitions the data using the Akaike or Bayesian Information Criterion, g-means [11] which tests the hypothesis that each cluster is generated from a Gaussian and pg-means [12] which extends g-means by testing all clusters jointly and allows for Gaussian clusters with different covariances.

V. Evaluation

Datasets and baselines For the experimental evaluation we use 10 datasets, 5 synthetic and 5 real-world, and 7 algorithms. The synthetic datasets are depicted in Figure 8. NORM consists of 35 well separated spherical Gaussian clusters (n=5000, m=2). OVER consists of 15 overlapping, non spherical Gaussian clusters (n=5000, m=2). COMP consists of two uniform clusters of complex shape (n=540, m=2). FADE consists of elongated fading clusters generated using the beta distribution (n=1500, m=2). SHAPE consists of 6 uniform clusters of various shapes and distances (n=788,m=2). Each datasets tests the algorithms' ability to discover a particular type of structure. NORM, OVER and SHAPE are taken from [18] (named as A2, S3 and aggregate). The real world datasets are: PEN which comprises 4x4 gray-scale images of handwritten digits (n=3498, m=16), IRIS which contains instances of various iris plants (n=150, m=4), AIR which contains flight passenger data regarding the number of local and foreign passengers for various airports (n=1584, m=2), BANK which contains banknote authentication data (n=1372, m=4) and SEG which contains various statistics concerning image segments of objects (n=2000, m=20). In the real world datasets the class label of each instance is used to indicate cluster membership. PEN, IRIS, BANK and SEG are taken from the UCI repository [19] and AIR is taken from Kaggle. Datasets NORM, OVER, PEN and IRIS are well known benchmark datasets. The size and dimensions of the datasets are inline with related work. All datasets and the code of our implementation of M-Dip are available at http://tiny.cc/0zps7y.


(a) NORM (b) OVER

(c) FADE (d) COMPD (e) SHAPE

Fig. 8: Synthetic datasets

	M-DIP	SKN	DM	HDB	ADP	GAP	XM	GM
NORM (35)	32	25	32	34	31	5	35	35
OVER (15)	15	16	10	25	15	2	1	148
COMP (2)	2	2	6	2	10	4	2	10
FADE (3)	3	3	6	4	4	2	57	36
SHAPE (7)	5	4	5	5	4	3	1	9

TABLE II: Number of clusters

	M-DIP	SKN	DM	HDB	ADP	GAP	XM	GM
NORM	0.97	0.85	0.96	0.85	0.94	0.58	**0.99**	**0.99**
OVER	**0.79**	0.77	0.72	0.53	**0.79**	0.33	0.00	0.62
COMP	**1.00**	0.45	0.52	1.00	0.46	0.00	0.22	0.46
FADE	**0.93**	0.68	0.71	0.85	0.74	0.33	0.49	0.45
SHAPE	**0.88**	0.79	0.87	**0.88**	0.82	0	0.77	0.87
AIR	**0.75**	0.66	0.64	0.63	0.59	0.00	0.0	0.46
PEN	**0.78**	0.63	0.59	0.73	0.69	0.60	0.54	0.68
IRIS	**0.73**	0.65	0.65	**0.73**	**0.73**	0.69	0.59	0.70
BANK	**0.45**	0.43	0.15	0.22	0.05	0.00	0.28	0.29
SEG	0.61	0.56	0.42	**0.62**	0.32	0.0	0.46	0.52
AVG-SN	**0.91**	0.70	0.75	0.82	0.75	0.24	0.49	0.67
AVG-RL	**0.66**	0.58	0.49	0.58	0.47	0.25	0.37	0.53

TABLE III: Clustering quality (NMI)

	M-DIP	SKN	DM	HDB	ADP	GAP	XM	GM
NORM	0.91	0.56	0.84	0.56	0.84	0.19	**0.99**	**0.99**
OVER	**0.72**	0.67	0.51	0.11	**0.72**	0.10	0.00	0.20
COMP	**1.00**	0.55	0.32	1.00	0.20	0.00	0.29	0.20
FADE	**0.94**	0.70	0.61	0.89	0.67	0.30	0.26	0.26
SHAPE	**0.80**	0.73	0.82	**0.80**	0.77	0	0.45	0.69
AIR	**0.60**	0.55	0.46	0.55	0.46	0.00	0.0	0.09
PEN	**0.66**	0.48	0.34	0.53	0.53	0.55	0.04	0.28
IRIS	0.56	0.53	0.53	0.56	0.56	**0.73**	0.34	0.61
BANK	**0.32**	0.31	0.12	0.08	0.06	0.00	0.01	0.03
SEG	**0.48**	0.35	0.23	0.44	0.10	0	0.01	0.14
AVG-SN	**0.87**	0.64	0.62	0.67	0.64	0.11	0.39	0.46
AVG-RL	**0.52**	0.44	0.33	0.43	0.34	0.25	0.08	0.23

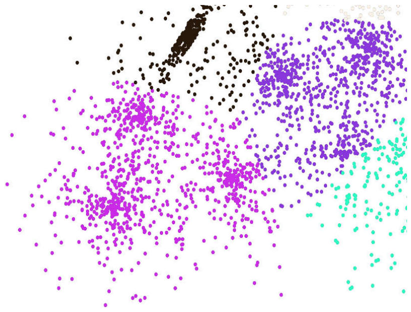
TABLE IV: Clustering quality (ARI)

For comparison we use the following algorithms, described in Section IV: skinny-dip (SKN) [1], dip-means (DM) [2], hdbscan (HDB) [4], the adaptive density peak detection algorithm from [3] (ADP), gap algorithm (GAP) [17], x-means (XM) [10] and g-means [11]. We focus mostly on SKN and DM, upon which we attempt to improve. All algorithms were used with the default parameters provided in their implementations. M-DIP was used with $\alpha = 0.95$ and $k = 30$, as suggested by the analysis of Section III-B. For the evaluation we use three metrics: normalised mutual information (NMI), adjusted rand index (ARI) and the number of clusters, for the synthetic datasets, where the number of clusters is known.

Results Tables II, III, IV and II present the results. M-DIP achieves the highest average NMI and ARI scores, by a significant margin over the second best algorithm, on both the synthetic (relative improvement: 11% NMI, 30% ARI) and real world datasets (relative improvement: 13% NMI, 18% ARI). It also achieves the highest score in most datasets individually. SKN achieves the second best average score in the real world datasets and HBD is second in the synthetic datasets. The performance of DM is average overall. Its good performance on NORM and OVER suggests it requires the existence of Gaussian clusters. This is justified by its usage of k-means, to assign points to clusters. Similarly XM and GM, which assume Gaussian clusters perform well on NORM, but they do not perform very well on other datasets. Notably, ADP and HDB perform consistently well on many datasets. The discrepancy between NMI and ARI scores for HDB suggests it generates many false positive clusters (e.g., OVER). Finally, GAP does not perform well overall, as it tends to underestimate the number of clusters, but has the best score in one case (ARI on IRIS dataset). Regarding the number of discovered clusters (Table II, true number of clusters in parentheses), M-DIP identifies the correct number on three out of five datasets and slightly underestimates the number of clusters on NORM and SHAPE dataset. In comparison, SKN correctly estimates the number of clusters in two datasets and DM does not find the correct the number in any dataset.

The advantages of searching for dips directly in the multidimensional space is illustrated in the examples of Figures 10 and 9. Figure 10 depicts the clusters discovered by M-DIP and SKN on a location near the center of dataset NORM. The clustering was performed in the entire dataset but only a small location is depicted, for clarity. As we can see, in this case, the linear decision boundaries found by SKN can not effectively discover the cluster number and the cluster boundaries. This occurs because clusters that do not overlap in the original space may overlap in the linear projections, resulting in decision bounds that run through parts of multiple clusters, (Figure 10a). On opposite, by considering the multidimensional space directly, M-DIP can correctly discover the clusters and their boundaries (Figure 10b). Although this is illustrated in a specific example, the results suggest that it occurs in other datasets as well. Even in COMP and FADE where SKN correctly estimates the number of clusters, their boundaries are not correct, as suggested by ARI and NMI.

Figure 9 presents a similar example for DM, taken from dataset OVER, from the bottom left part of the dataset. In this case neighboring, overlapping clusters are merged together by DM Figure 9a, instead of being identified as separate. It is not clear if this is a limitation in detecting the existence of a new cluster or at correctly discovering their boundaries (or both). In


(a) DM

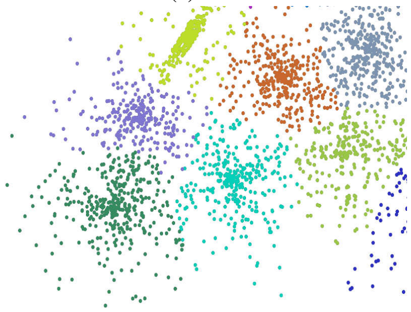
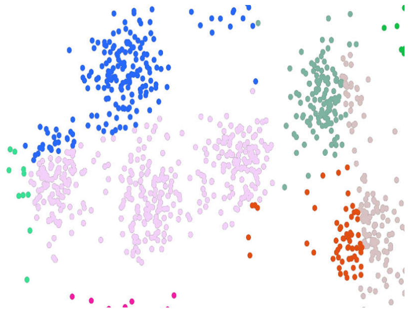

(b) M-DIP

Fig. 9: Discovered clusters on an indicative sample of OVER dataset, for M-DIP and DM


(a) SKND

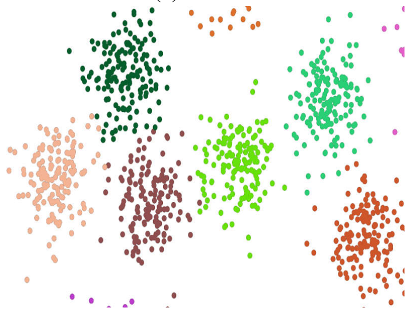

(b) M-DIP

Fig. 10: Discovered clusters on an indicative sample of NORM dataset, for M-DIP and SKND

either case, by detecting the dip in the multidimensional space directly, M-DIP correctly detects the exiting clusters and their boundaries.

VI. CONCLUSION

Clustering algorithms are used to find groups of similar items in a dataset. Automatic clustering algorithms achieve this task without requiring users to input critical parameters. In this paper we presented M-DIP, an automatic clustering algorithm which generalises the density dip search methodology of [1] and [2] in the multi-dimensional space. M-DIP works by measuring the density dips on a dataset and comparing them with a threshold of statistical significance, which it calculates through simulations. In our experimental evaluation, M-DIP achieves better results that the existing state of the art in automatic clustering. Directions for improvement include studying the effects of k-NN graph's connectivity in calculating the threshold threshold of statistical significance θ, and investigating the use of distance instead of density, which may result in a more stable metric.

ACKNOWLEDGEMENTS

This work was funded by the EU H2020 projects SLIPO (731581) and SmartDataLake (825041) and the the NSRF 2014-2020 project HELIX (5002781).

REFERENCES

[1] S. Maurus and C. Plant, "Skinny-dip: Clustering in a sea of noise," *KDD*, 2016.

[2] A. Kalogeratos and A. Likas, "Dip-means: an incremental clustering method for estimating the number of clusters," *NIPS*, 2012.

[3] X. Wang and Y. Xu, "Fast clustering using adaptive density peak detection," *Journal of Statistical Methods in Medical Research*, 2017.

[4] R. Campello, D. Moulavi, and J. Sander, "Density-based clustering based on hierarchical density estimates," *PAKDD*, 2013.

[5] J. Hartigan and P. Hartigan, "The dip test of unimodality," *The Annals of Statistics*, 1985.

[6] J. Chen, H. Fang, and Y. Saad, "Fast approximate knn graph construction for high dimensional data via recursive lanczos bisection," *JLMR*, 2009.

[7] C. Bhm, C. Faloutsos, J. Pan, and C. Plant., "Robust information-theoretic clustering," *KDD*, 2006.

[8] C. Bhm, C. Plant, J. Shao, and Q. Yang, "Clustering by synchronization," *KDD*, 2010.

[9] L. Zelnik-Manor and P. Perona, "Self-tuning spectral clustering," *NIPS*, 2004.

[10] D. Pelleg and A. Moore, "X-means: Extenking k-means with efficient estimation of the number of clusters," *ICML*, 2000.

[11] G. Hamerly and C. Elkan, "Learning the k in k-means," *NIPS*, 2003.

[12] Y. Feng and G. Hamerly, "Pg-means: learning the number of clusters in data," *NIPS*, 2006.

[13] Y. Benjamini and Y. Hochberg, "Controlling the false discovery rate: a practical and powerful approach to multiple testing," *Journal of the Royal Statistical Society*, 1995.

[14] A. Rodriguez and A. Laio, "Clustering by fast search and find of density peaks," *Science*, 2014.

[15] G. Wang and Q. Song, "Automatic clustering via outward statistical testing on density metrics," *TKDE*, 2016.

[16] P. Rousseeuw, "Silhouettes: a graphical aid to the interpretation and validation of cluster analysis," *Journal of Computational and Applied Mathematics*, 1987.

[17] R. Tibshirani, G. Walther, and T. Hastie, "Estimating the number of clusters in a data set via the gap statistic," *Journal of the Royal Statistical Society*, 2000.

[18] P. Fränti and S. Sieranoja, "K-means properties on six clustering benchmark datasets," 2018. [Online]. Available: http://cs.uef.fi/sipu/datasets/

[19] D. Dheeru and E. Karra-Taniskidou, "UCI machine learning repository," 2017. [Online]. Available: http://archive.ics.uci.edu/ml