Automatic topography of high-dimensional data sets by non-parametric density peak clustering

Maria d'Errico [a,b,c], Elena Facco [a], Alessandro Laio [a,d], Alex Rodriguez [a,d]

[a] SISSA, Scuola Internazionale Superiore Studi Avanzati, via Bonomea 265, I-34136 Trieste, Italy
[b] Functional Genomics Center Zurich, UZH/ETHZ, 8057 Zurich, Switzerland
[c] Swiss Institute of Bioinformatics, Quartier Sorge – Batiment Amphipole 1015 Lausanne, Switzerland
[d] ICTP, International Centre for Theoretical Physics, Strada Costiera 11, I-34100 Trieste, Italy

ARTICLE INFO

ABSTRACT

Data analysis in high-dimensional spaces aims at obtaining a synthetic description of a data set, revealing its main structure and its salient features. We here introduce an approach providing this description in the form of a *topography* of the data, namely a human-readable chart of the probability density from which the data are harvested. The approach is based on an unsupervised extension of Density Peak clustering and on a non-parametric density estimator that measures the probability density in the manifold containing the data. This allows finding automatically the number and the height of the peaks of the probability density, and the depth of the "valleys" separating them. Importantly, the density estimator provides a measure of the error, which allows distinguishing genuine density peaks from density fluctuations due to finite sampling. The approach thus provides robust and visual information about the density peaks height, their statistical reliability and their hierarchical organization, offering a conceptually powerful extension of the standard clustering partitions. We show that this framework is particularly useful in the analysis of complex data sets.

© 2021 Elsevier Inc. All rights reserved.

1. Introduction

The rapidly increasing capability to generate data calls for approaches able to provide a compact representation of their underlying structure. The challenge is to extract from data sets with, say, 1000 dimensions an information content at the same time human readable and useful.

A possible route to achieve this goal is attempting to map the data on a two or three dimensional surface that can then be directly visualized. This low-dimensional representations of high-dimensional data can be derived, for example, using Principal Component Analysis [36] and, within a framework which allows taking non-linearities into account, by multidimensional scaling [47], ISOMAP [46], Diffusion Maps [9], Locally Linear Embedding [39], Tree Preserving Embedding [41], t-Distributed Stochastic Neighbor Embedding [27] and Sketch-Map [6]. However, the intrinsic dimensionality (ID) of realistic data sets is often larger than three. This has become more and more evident in recent years, thanks to the development of powerful and accurate approaches capable of estimating the ID [4,18,12]. If the ID of a data set is, say, 10 any attempt to describe it with only two or three coordinates unavoidably leads to an information loss. This can lead to several effects like altering the local neighborhood structures of the data sets, or the crowding of the data due to the reduction of space when

---

passing from high dimension to low dimension [17]. Most of the above-mentioned methods try to quantify the amount of information preserved in the projection by a suitable objective function. For instance, in Multidimensional Scaling the objective function measures the preservation of the distances, while in t-SNE the information preserved is the neighborhood structure [3]. However, this information loss can lead, depending on the data set, to the presence of artifacts that make those methods not applicable for a quantitative analysis in case of large IDs, although they may be still useful as preprocessing, or for visualization purposes. One example (among many) of these problems can be found in Ref. [45], where ISOMAP was employed for projecting a folding trajectory of Villin protein described by the 32 features (with an ID of 12). In this particular case the projection was not able of distinguish folded or unfolded states nor provide a useful visualization of the data set.

A different strategy for summarizing the information content of a data set is considering the data as an ensemble of realizations drawn from a probability distribution where the regions of space with higher density of data points, generally defined as clusters, correspond to probability peaks. Density-based clustering [11,10,38] allows finding those peaks and estimating their properties without projecting the data onto a lower-dimensional representation. This approach offers two major advantages. First, it can be followed even in the case of high intrinsic dimensionalities. Second, it can also be exploited to formulate a hierarchical representation of the probability distribution, by establishing a hierarchy of connected subsets of points. This idea was introduced in the seminal work of Hartigan [19], and has been exploited in many recent algorithms like, for instance, HDBSCAN [5], Robust Single Linkage [8] and Robust Density-Based Clustering [44].

In this work we introduce Density Peaks Advanced (DPA), a method for reconstructing what we call the *topography* of a data set, or a simplified human-readable chart of the probability distribution. The topography conveys information on the height of all the probability peaks as well as on the organization of these peaks in larger structures. When density peaks are identified as clusters, the topography provides an immediate visual information about their relationships. If the probability distribution includes *N* peaks (or clusters), the topography consists in a $N \times N$ symmetric matrix in which the diagonal entries are the heights of the peaks and the off-diagonal entries are the heights of the saddle points. As we discuss in the following, a saddle point between two peaks is estimated by searching the point of highest density among all the points at the border between the two peaks. An off-diagonal entry is set to zero if the two peaks are not in contact. This matrix can be represented in the form of a tree diagram, like in Refs. [19,5,8,44], obtaining a chart that unveils the hierarchies by focusing on the highest saddles between peaks. We will also show that complementary information about the topography can be visualized by applying one of the approaches developed for representing the kinetic models derived by Markov State Model analysis [34].

The topography is reconstructed by using a modified version of the Density Peaks (DP) algorithm [38]. This approach provides an empirical criterion for a quick and reliable localization of density peaks. The original formulation of DP is affected by two main drawbacks. First, the selection of cluster centers is relatively subjective, since it is based on the visual inspection of the so-called decision graph: a scatter plot of the density of a point vs its minimum distance from a point with higher density (see Ref. [38]). Second, like all density-based clustering approaches, it is sensitive to the parameters involved in the density estimation [50]. These drawbacks have been addressed in many works. For instance, Ref. [26] automatically finds the number of clusters through a recursive inspection of the decision graph based on a Divide-and-Conquer strategy. In the two-dimensional case, Ref. [29] introduces instead a non-parametric technique for estimating the densities based on the heat diffusion equation. Although the method in Ref. [29] still requires the inspection of the decision graph, it shows an improved performance in the classification of artificial data sets.

In this work we demonstrate that the DP clustering algorithm can be made fully unsupervised by combining it with PA*k* [37], a non-parametric density estimator recently proposed by us. This estimator exploits a statistical approach to find the largest region around each point in which the density is approximately constant. One of the PA*k*'s main innovations with respect to other non-parametric estimators is that it measures the density in the manifold in which the data lay, and not in the embedding space whose dimensionality is normally overwhelmingly large. In the following we show that the mathematical formulation of the estimator in Ref. [37] naturally induces a criterion to automatically find density peaks through DP clustering. Moreover, the estimator provides a measure of the uncertainty on the density. This last feature is a key ingredient, since it allows recognizing genuine density peaks from statistical fluctuations of the estimated density due to finite sampling. In fact, our approach allows assessing the statistical reliability of probability peaks thus providing a new manner for performing a so-called multimodality test (see, for instance, Refs. [42,30]). This, to the best of our knowledge, is an original contribution of this work.

After testing the DPA algorithm in several toy problems, we analyze two real world data sets: the MNIST database [24] of handwritten digits and a sample of protein sequences extracted from the Pfam clan PUA [14,15], a complex superfamily of sequences organized into ten families, each containing a variety of architectures.

2. Methods

2.1. Topography of probability density function landscapes

Data sets can often be described as realizations of an underlying probability distribution whose density has support in the space of the features (coordinates) of the data. This density can be characterized by the presence of several maxima, at times organized hierarchically, as in the synthetic example shown in Fig. 1A.
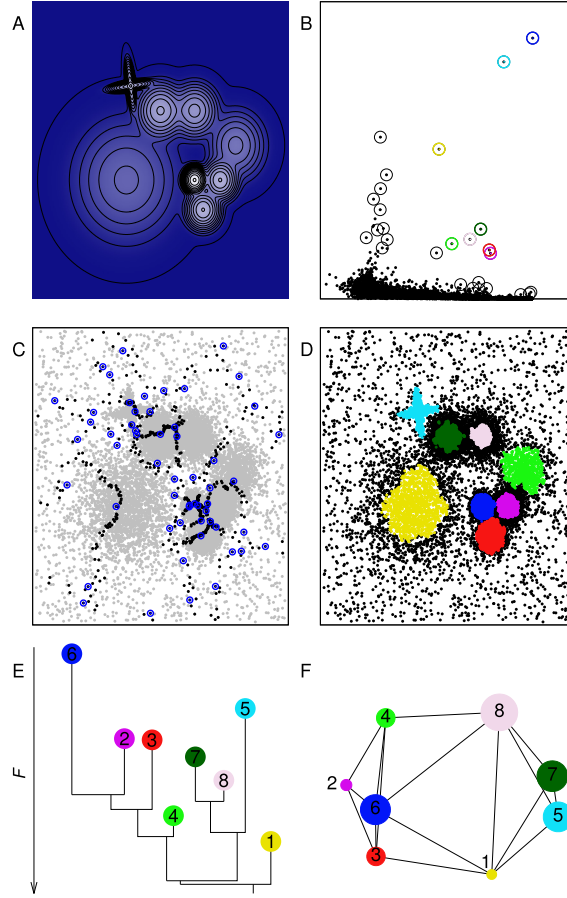
Fig. 1. CLUS8, a two dimensional toy example. (A) Probability density function from which the 20000 points have been drawn. (B) Decision graph using *g* defined in Eq. (3) as density. Circled points correspond to the putative centers chosen according with Heuristic 1, while the colored ones are the centers of the ground truth clusters. (C) Borders (black points) and saddles (blue points) between clusters. (D) Final assignation of the points to clusters, black points correspond to halo points. The color code is the same for Panels B, E and F. (E) Dendrogram representation of the data set topography. (F) Network representation of the data set topography. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

The scope of our approach is reconstructing with no supervision the topography of those complex probability distributions. A key step of this procedure consists in the identification of the density maxima within the data set and the saddle points between them.

We will show that the density peaks and saddle points can be automatically recognized by making use of the PA*k* (Point Adaptive *k*-nearest neighbor) density estimator [37] within the framework of Density Peaks clustering [38].

2.1.1. An adaptive k-Nearest Neighbor density estimator

In this section we will describe in short the PA*k* density estimator as introduced in Ref. [37]. PA*k* aims to estimate the local density around each point *i* in a data set. We denote by $\{r_{i,l}\}_{l\leqslant k}$ the sequence of the ordered distances between *i* and its first *k*-nearest neighbors, and by $v_{i,l} = \omega\left(r_{i,l}^d - r_{i,l-1}^d\right)$ the volumes of the hyperspherical shells enclosed between two successive neighbors $l-1$ and *l*, where *d* is the Intrinsic Dimension (ID) of the manifold in which the data points lay. As we discussed in Ref. [37] the density should be estimated by measuring the volumes in the embedding manifold whose dimension is *d*, rather than the extrinsic dimension of the space in which the data points are defined, which can be orders of magnitude larger than *d*. The value of *d* can be estimated using one of the many approaches for computing the intrinsic dimension. We here estimate it by the TWO-NN method [12].

PA*k* relies on the observation that, for a given point *i*, if the density is constant the volumes $v_{i,l}$ are independently drawn from an exponential distribution with rate equal to the density ρ, as proven in Ref. [12]. Therefore, the log-likelihood function of the parameter ρ given the observation of the *k*-nearest neighbor distances from point *i* is

$$\mathscr{L}_{i,k}(\rho_i) = k \cdot \log \rho_i - \rho_i \cdot \sum_{l=1}^{k} v_{i,l} = k \cdot \log \rho_i - \rho_i \cdot V_{i,k}. \tag{1}$$

where $V_{i,k}$ denotes the total volume occupied by the k nearest neighbors of point i.

Indeed, the maximization of Eq. (1) with respect ρ_i leads to the k-Nearest Neighbor density estimator with an associated variance that decreases with the square root of k, so the estimate improves when using high values of k. However, when k increases, the density in the neighborhood within a distance $r_{i,k}$ from the data point i can become non-constant, breaking the hypothesis from which Eq. (1) has been derived and therefore inducing systematic errors (bias). PAk finds, for each data point, the largest neighborhood $\hat{k}$ at which the density can be considered constant by performing a Likelihood Ratio Test [31] between two models: (1) a model in which the densities at the point and at its kth nearest neighbor are considered the same and (2) a model in which the densities are assumed to be different. This test involves estimating the difference D_k of the log-likelihood of the two models for increasing values of k. For each point i the algorithm chooses the optimal value of k, denoted by $\hat{k}$, according to the condition

$$\hat{k}_i : (D_k < D_{\text{thr}} \ \forall \ k \leqslant \hat{k}_i) \& (D_{\hat{k}_i+1} \geqslant D_{\text{thr}}) \tag{2}$$

where $D_{\text{thr}} = 23.928$, corresponding to a p-value of 10^{-6}. This implies that for point i the log-likelihoods of the two models are consistent within a p-value of 10^{-6} for $k \leqslant \hat{k}_i$.

Once the optimal k for point i has been found, the algorithm corrects for the systematic errors induced by choosing $\hat{k}_i$ according to a fixed confidence threshold by modifying the log-likelihood function in Eq. (1) to include an extra variational parameter a. This parameter describes the linear trend in the density estimation as one moves further and further from the central point. At this point the density ρ_i and the corresponding uncertainty ϵ_i are obtained by maximizing the log-likelihood in Eq. (1) with k equal to $\hat{k}_i$ and $\log(\rho)$ replaced by $\log(\rho) + a \cdot l$. By leveraging Fisher information, the variance of $\log(\rho)$ can be derived as $\varepsilon_i = \sqrt{\frac{4 \cdot \hat{k}_i + 2}{(\hat{k}_i - 1) \cdot \hat{k}_i}}$. In Supp. Inf. Text S1 we provide the derivation of this expression and a pseudocode for obtaining the PAk estimation. More details and a validations of the procedure can be found in Ref. [37].

2.1.2. Automatic detection of density peaks

The first step of our algorithm is finding automatically the density peaks. As in the standard Density Peaks (DP) clustering [38], we here assume that the density peaks are surrounded by neighbors with lower local density and that they are at a relatively large distance from any points with a higher local density. However, while in the standard DP algorithm this definition is left to the interpretation of the so-called decision graph, in this section we show that the additional information provided by PAk (an estimate of the error in the density and the neighborhood size around each point in which the density can be considered approximately constant) can be exploited to provide a quantitative definition of density peaks, therefore allowing their unsupervised detection.

To detect density peaks we do not directly consider the density of points, that typically varies by several orders of magnitude, but the logarithm of their density $\log(\rho_i) = -F_i$ identified as the free energy at point i from the PAk estimator [37]. As it is, the estimate of F_i is affected by non-uniform errors, and defining cluster centers as the maxima of the density as implicitly done in in the DP clustering [38] is not always effective. We indeed verified that a straightforward combination of DP clustering with the PAk density estimator fails to properly localize the correct clusters when the error is highly non-uniform: a point whose estimated $\log(\rho_i)$ is large but affected by a comparatively large error ε_i is not likely to be a genuine density peak. To tackle this problem we developed a heuristic approach for defining cluster centers, which provides reliable results in practical applications, as we show in the following. We define as cluster centers the local maxima of g_i, where g_i is defined as

$$g_i = \log(\rho_i) - \varepsilon_i. \tag{3}$$

This definition is a generalization of the one used in Ref. [38], since the local maxima of g_i coincide with the local maxima of ρ_i if the error is uniform. If the error is not uniform, points with large error are less likely to be selected as local maxima with respect to points with a small error. Following Ref. [38], we then compute $\delta_i = \min_{j:g_j>g_i} r_{ij}$, namely the distance to the nearest point with higher g, and we automatically find the cluster centers using the following heuristic.

Heuristic 1. We consider point i a putative center if it satisfies the following property: all its $\hat{k}$ nearest neighbors contributing to determine the value of its density have a value of g lower than g_i. As a second condition a center can not belong to the neighborhood of any other point with higher g:

1. $\delta_i > r_{\hat{k}_i}$
2. $\forall \ j$ with $g_j > g_i : i \notin NN_j := \{k : r_{jk} < r_{\hat{k}_j}\}$

In this heuristic, $\hat{k}_i$ is the optimal number of nearest neighbors defined as in Ref. [37], and NN_j denotes the optimal neighborhood of j. Moreover, the second criterion in Heuristic 1 makes putative centers selection more robust in front of statistical fluctuations in the neighborhood estimation, see Fig. 2A for an illustrative example. A pseudocode implementing this heuristic is presented in Algorithm 1.

Algorithm 1: Automatic detection of cluster centers

Require: Set of points N_{points} $x_1, \ldots, x_N \in \mathbb{R}^d$
Ensure: *Centers*, the list of points identified as centers.

1. $\log(\rho), \epsilon, \hat{k} \leftarrow \text{PA}k(N_{\text{points}})$ ▷ where ρ-density, ϵ-error, $\hat{k}$-optimal neighborhood size
2. $g := \log(\rho) - \epsilon$ ▷ compute g for all points in N_{points}
3. *Centers* ← empty array
4. **for** $i \leftarrow 1, N$ **do**   ▷ (criterion 1 in Heuristic 1)
5. *putative_center* ← True
6. **for** $j \leftarrow 1, \hat{k}_i$ **do**   ▷ loop over the points in NN_i, the neighborhood of i
7. **if** $g_j > g_i$ **then**
8. *putative_center* ← False
9. **break**
10. **if** *putative_center* = True **then**   ▷ the closest point with higher density is not in NN_i, thus Criterion 1 is satisfied.
11. add i to *Centers*
12. **for** i in *Centers* **do**   ▷ (criterion 2 in Heuristic 1)
13. **for** $j \leftarrow 1, N$ **do**
14. **if** $g_j > g_i$ and $j \in NN_i$ **then**
15. remove i from *Centers*   ▷ i belongs to the neighborhood of another point with higher g

---

In Fig. 1B we show the decision graph (i.e., the value of δ_i as a function of g_i) for a sample of 20000 points extracted from the probability density distribution shown in Panel A. The points surrounded by a circle are those that are automatically chosen as putative centers according to Heuristic 1.

The next step is to assign all the points that are not centers to the same cluster as the nearest point with higher g. This assignation is performed in order of decreasing g. Choosing the points highlighted in Fig. 1B as centers leads to a high splitting of the data set (see Figure S1 for the result of this preliminary assignation). Indeed, Heuristic 1 correctly identifies the genuine probability peaks but also the spurious statistical fluctuations of the density induced by the finite sampling. Consequently, we developed a protocol to assess the peaks significance, which allows distinguishing meaningful density peaks from statistical fluctuations of the density, as explained in the following sections.

2.1.3. Finding the saddle points

We here introduce a procedure that allows finding the saddle points of the probability density function between each peak and its neighboring ones, which will be crucial for providing insights on the structure of the data. We first find the data points that are at the border between two clusters following Heuristic 2.

Heuristic 2. A point i belonging to cluster c is assumed to be at the border between cluster c and c' if its closest point j belonging to c' is within a distance $r_{\hat{k}_i}$ and if i is the closest point to j among those belonging to c:

1. Let $j = \arg\min_{k:k \in c'} r_{ik}, r_{ij} < r_{\hat{k}_i}$
2. $i = \arg\min_{k:k \in c} r_{jk}$

The saddle point of the probability density function between a pair of clusters c and c' is defined as the point with the highest value of g among those at the border between c and c', see Fig. 2B for an illustrative example. The value of the logarithm of the density of this point and its error are denoted by $\log \rho_{cc'}$ and $\varepsilon_{cc'}$. The border points between the clusters reconstructed from the two dimensional example in Fig. 1 are shown in black in Panel C, while the saddle points are circled in blue. The pseudocode implementing this heuristic is presented in Algorithm 2.

Algorithm 2: Saddle points between clusters c and c'

Require: Neighborhood size $\hat{k}$ and the cluster label assigned to each
    point in N_{points}
Ensure: $Border_{cc'}$, the list of points at the border between c and c'.
1. $Border_{cc'} \leftarrow$ empty array
2. **for** i in c **do**   ▷ (criterion 1 in Heuristic 2)
3. **for** $j \leftarrow 1, \hat{k}_i$ **do**   ▷ loop over the points in NN_i, the
                                                      neighborhood of i j c'
4. **if** $j$inc' **then**
5. $candidate \leftarrow j$
6. **break**
7. **for** $k \leftarrow 1, \hat{k}_j$**do**   ▷ (criterion 2 in Heuristic 2)
8. **if** $k = i$ **then**
9. $Border_{cc'} \leftarrow candidate$
10. **break**
11. **if** k in c: **then**
12. **break**   ▷ i is not the closest to j

2.1.4. Assessing the peaks and assignation significance

Based on the value of $\log(\rho_{cc'})$, the logarithm of the density at the saddles, and their error we introduce a criterion for distinguishing genuine density peaks from statistical fluctuations of the density due to finite sampling, as defined by Heuristic 3.

Heuristic 3. A cluster c is considered as the result of a statistical fluctuation if all the points assigned to it have density values compatible, within their errors, with the border density. The cluster is thus merged with a neighboring cluster c' if:

1. $(\log \rho_c - \log \rho_{cc'}) < Z \cdot (\varepsilon_c + \varepsilon_{cc'})$.

where ρ_c is the density of the center of cluster c.

The constant Z entering Heuristic 3 fixes the level of statistical confidence at which one decides to consider a cluster meaningful. It is the only free parameter of our approach, but its value has a clear statistical interpretation (see Fig. 2C for an illustrative example). Heuristic 3 is checked for all the clusters c and c' in order of decreasing $\log \rho_{cc'}$. Therefore, the implemented procedure performs a multimodality test based in the error estimated with PAk, which iteratively compares the difference between the estimated log-density at each pair of peaks (clusters) and the log-density estimated at the saddle point between them. This procedure prunes the set of clusters from those corresponding to density maxima that are not statistically robust, thus recovering the topography of the underlying probability function.

Furthermore, the knowledge of the border densities between clusters allows to identify the set of points whose assignation is not reliable [38]. Indeed, DPA inherits from standard DP clustering the capability of classifying the points as 'core' (those whose assignation is robust) or 'halo' (they are assigned to a cluster but with a lower level of confidence). Points are classified as 'halo' if their density is lower than the highest border density between the cluster to which they are assigned and any other cluster.

The pseudocode implementing this heuristic is presented in Algorithm 3.

Algorithm 3: Assessing peaks significance

Require: $Border_{cc'}$, the list of points at the border between c and c'.
Ensure: $Topography$: $N_{clus} \times N_{clus}$ symmetric matrix, in which the diagonal entries are the
    height of the peaks and the off-diagonal entries are the heights of the saddle points.
    clu_lables: the final assignation to clusters after merging. $halos$: points with not reliable
    assignation.
1. $apply\_merging \leftarrow$ True
2. **while** $apply\_merging$ is True **do**
3. $apply\_merging \leftarrow$ False

(*continued on next page*)

4. **for** points in $Border_{cc'}$ **do**
5. **if** Condition (3) is True **then**   ▷ (criterion 1 in Heuristic 3)

6. *apply merging ← True*
7. **if** *apply_merging* is True **then**
8. $(c_h, c_l) \leftarrow$ pair with the highest $\rho_{c,c'}$ ▷ c_h is the peak with highest ρ

9. merge c_l with c_h
10. update *Topography*
11. update *clu_labels*
12. *halos* ← empty array   ▷ Find halo points
13. **for** i in N **do**   ▷ loop over the point in the data set

14. **if** $\rho_i < \min\{\rho_{c,c'}\}$ **then** i
15. Add i to *halos*

In Fig. 1D it can be seen that the cluster assignment after the merging (with $Z = 1.5$) resembles almost perfectly the peaks shown in Panel A (black points correspond to halo). Indeed, these results will correspond with those obtained by the standard Density Peaks method if by visual inspection one chooses as centers the colored circles in Panel B.

2.1.5. Representing the topography

The information about the location and the height of the saddles allows building a compact representation of the topography of the probability distribution function from which the data points are harvested. To visualize the topography of the density distribution we follow two paths. Both are based on the fact that the higher the density of the border between clusters, the more similar such clusters can be considered. Therefore, we define the distance between two clusters as follows: $d_{cc'} = \max(\log(\rho_i)) - \log(\rho_{cc'})$. One possible way to visualize the topography is constructing a hierarchical tree by applying the Single Linkage algorithm [20] using $d_{cc'}$ as distance. This representation is similar to the one used in hierarchical density based methods [19,5,8,44]. In our case, to encode more information when representing the tree, the height of the branches is proportional to the density of the peak associated to them and the separation between branches in the x-axis is proportional to the population of the clusters. An example of this representation is provided in Fig. 1E.

An alternative way to represent the topography is by projecting the clusters in two dimensions and visualizing their relationship as a network where the thickness of the links between clusters is proportional to the log-density at the border. This leads to a representation similar to those used in Markov State Model analysis [34]. An example of this representation is provided in Fig. 1F. To encode more information in a single plot, the area of the circles representing the clusters is proportional to their population.

Both visualizations provide complementary information about the underlying probability density function. In the example shown in Fig. 1 the hierarchical relationship between clusters 2, 3 and 6 (magenta, red and blue) is more evident in the tree representation. However, the close contact between clusters 4 and 8 (light green and light pink) is evident only using the network representation. Additional examples are shown in Supp. Inf. (Fig. S2-5).

2.1.6. Statistical significance of the clusters

The value of Z in the Heuristic 3 is used to control the statistical reliability of the density peaks.

In general, at low Z values the method is more sensitive to variations of the density, but fluctuations due to sampling artifacts are also identified as clusters density. Then, the higher the value of Z, the lower the sensitivity to density changes, but the higher the statistical reliability of the peaks. If the sampling of the probability distribution function is good enough, one can increase the value of Z in order to enhance the statistical confidence. If the sampling is poor (something that easily happens if the intrinsic dimension of the data is high), one is forced to accept a lower level of confidence (lower value of Z) and a significant probability of observing some spurious clusters. In Fig. 3 we show an example of how the parameter Z affects the clustering classification in a well sampled distribution. Although the best results are obtained with $Z = 3$, and the number of clusters increases for smaller Z, the topography of the data set allows identifying the two main peaks of the distribution at any value of the parameter Z. A systematic study of the role of this parameter and its impact in detecting spurious clusters is provided in Supp. Inf. Text S1 and Fig. S6.

3. Results and discussion

3.1. Validation on artificial data sets

We first validate our approach on artificial two-dimensional data sets where the ground-truth classification is available. We perform benchmarks on the sets represented in Fig. 1 (hereafter called CLUS8), in Fig. 3 (SPIR2), and in Figure S2 (AGGR),
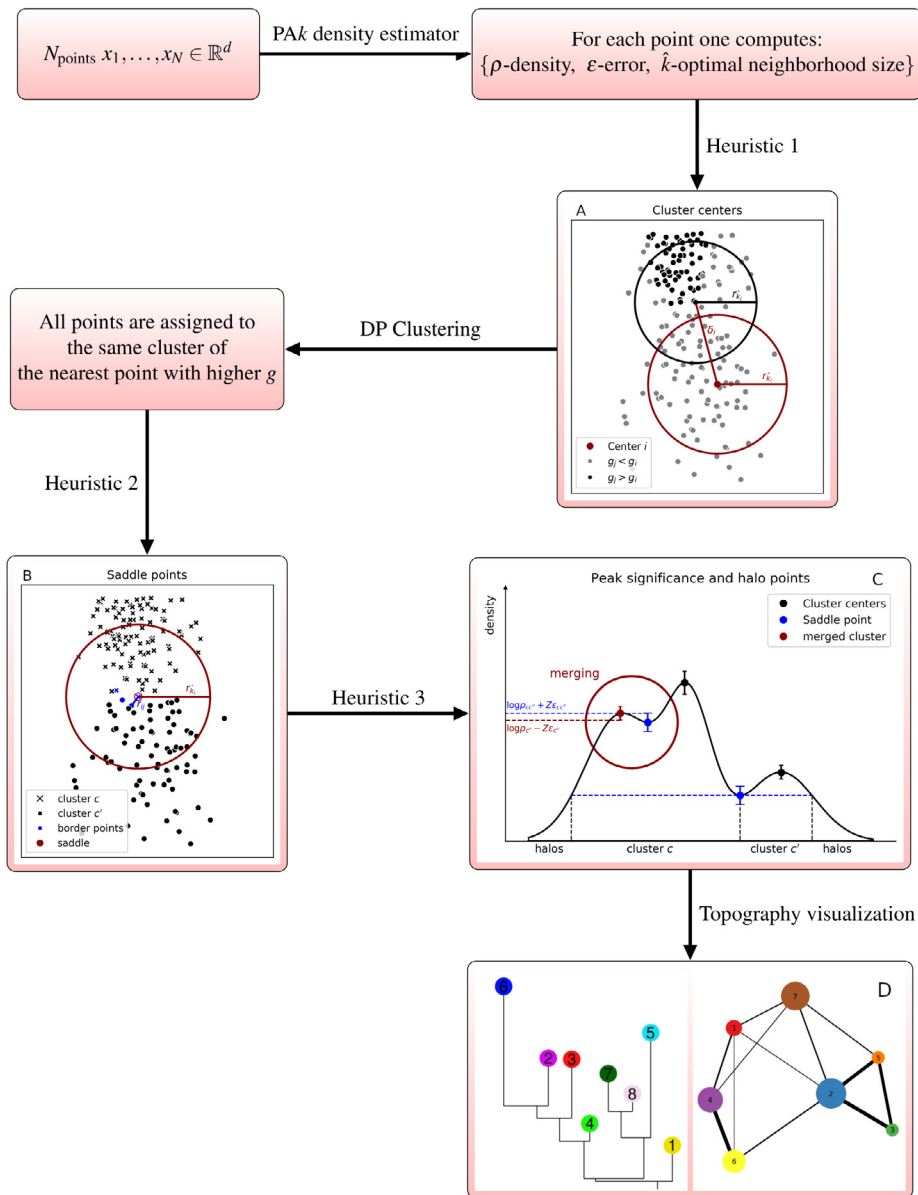
Fig. 2. Flowchart representation of the algorithm described in this work. The novel procedures are illustrated using two dimensional data sets: (A) Automatic detection of cluster centers (Heuristic 1), (B) Saddle points between cluster c and c' (Heuristic 2), (C) Merging of the clusters induced by statistical fluctuation (Heuristic 3) and (D) Topography representation.

S3 (SPIR3) and S4 (HORSE) in Supp. Inf., which have a number of clusters ranging between two and eight. In all the cases the ground-truth classifications are obtained using the explicit probability density function from which the points are generated: the ground-truth clusters correspond to maxima in this function. In detail, for each maximum we find the region such that: (i) contains the maximum; (ii) its boundary is a contour level of the probability density function passing through a saddle point; (iii) no other saddle points, except the one on the boundary, are contained in it. Data points belonging to this region form a ground-truth cluster, while the points which do not belong to any of these regions are classified as 'noise'. The artificial data sets and the corresponding ground-truth classifications are provided as supplementary material.

To assess the similarity between the DPA clustering assignation and the ground-truth classifications we use the Normalized Mutual Information (NMI) [48]. Note that when the two sets of labels have a perfect one-to-one correspondence, the NMI is equal to one. Since in the ground-truth some points are classified as 'noise', we quantify the accuracy in two different ways. (i) We check the accuracy in the clustering assignation of the points that belong to the ground-truth density peaks: we classify all the points using the DPA algorithm without labelling any point as 'halo', and compute the NMI by excluding the 'noise' points, which is denoted in the following as NMI_{dpa}. (ii) We check the accuracy of all points assignation, including
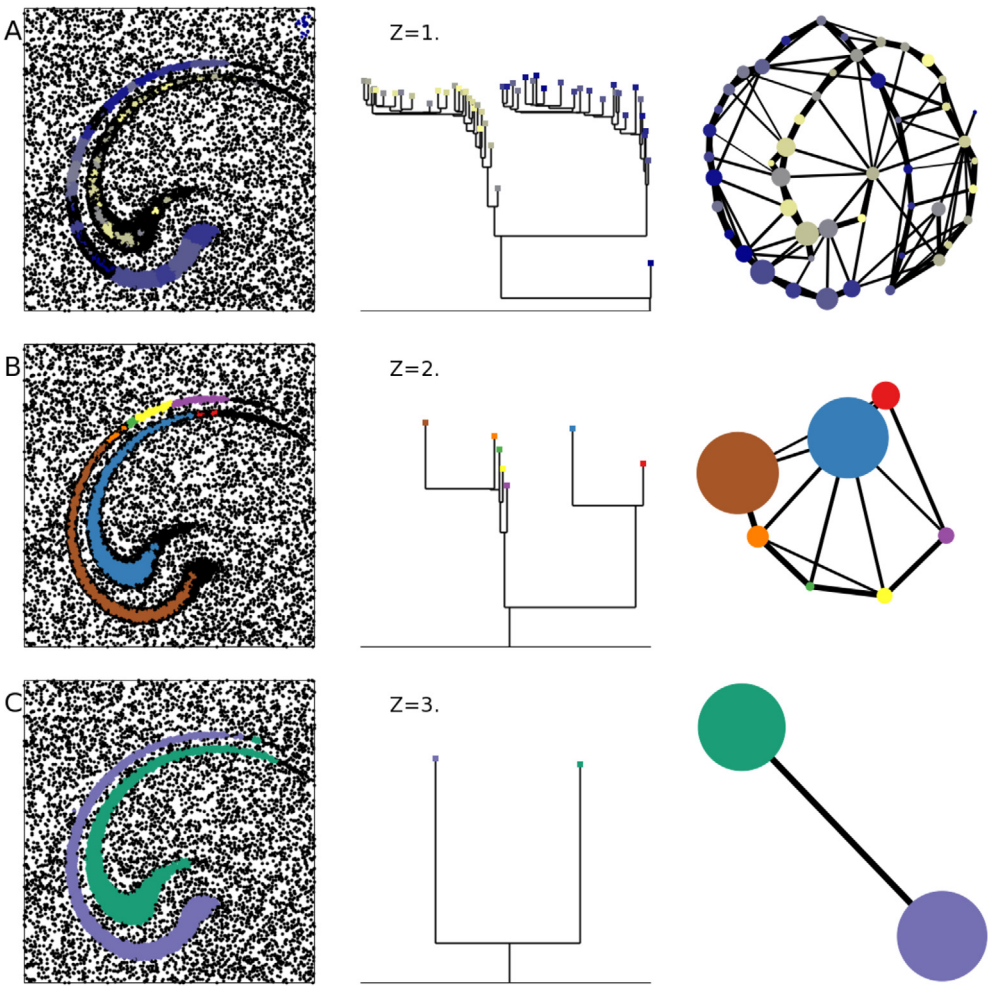
Fig. 3. Results of the topography reconstruction on SPIR2, a data set with two spirals, at different values of Z: $Z = 1$ (Panel A), $Z = 2$ (Panel B) and $Z = 3$ (Panel C). In each Panel, the colors in the assignation (left) are used for the corresponding branches in the dendrogram (middle) and circles in the network (right).

those classified as 'noise': by following the protocol described in Methods 2.1.4 we find the 'halo' points that should ideally correspond to the 'noise' classification, and then compute the NMI for all the points considering 'noise' as a ground-truth label, which is denoted in the following as NMI_{halo}. We also provide the False Negative and False Positive ratios for noise detection using halo points as noise classification (FNR_{halo} and FPR_{halo} respectively).

Table 1
Validation of the DPA clustering on artificial data sets. We use a selection of two-dimensional benchmark data sets having a number of clusters ranging between two and eight: data set CLUS8 represented in Fig. 1, SPIR2 in Fig. 3, and AGGR, SPIR3, and HORSE represented in Supp. Inf. in Figs. S2, S3 and S4 respectively. To assess the similarity between the DPA clustering assignation and the ground-truth classifications we measure the Normalized Mutual Information [48](NMI) in two different ways: NMI_{dpa} and NMI_{halo}. NMI_{dpa} is used to assess the accuracy in the clustering assignation of the points that belongs to the ground-truth density peaks, thus excluding the 'noise' points. NMI_{halo} is used to assess the accuracy of all points' assignation, by including those labelled as 'halo' (described in Methods 2.1.4) that should ideally correspond to the 'noise' classification. For each data set in the table we report $[Z_{best}]$ as the interval values of Z that maximize the NMI_{halo}, together with the corresponding value of NMI_{halo}. We report the interval $[Z_{90\%}]$ within which the NMI_{halo} is at least 90% of its maximum value, and N_{clus} as the number of clusters inferred for Z in $[Z_{best}]$. We also provide the False Negative and False Positive ratios for noise detection using halo points as noise classification (FNR_{halo} and FPR_{halo} respectively).

	$[Z_{best}]$	$[Z_{90\%}]$	N_{clus}	NMI_{dpa}	NMI_{halo}	FNR_{halo}	FPR_{halo}
CLUS8	[1.5, 1.9]	[1.1, 4.0]	8	0.996	0.841	0.010	0.111
SPIR2	[2.9, 4.0]	[2.9, 4.0]	2	0.966	0.929	0.002	0.015
AGGR	[2.0, 2.5]	[1.6, 2.8]	7	0.994	0.844	0.070	0.053
SPIR3	[2.5, 3.6]	[2.5, 3.6]	3	0.996	0.809	0.000	0.100
HORSE	[2.6, 4.0]	[2.6, 4.0]	3	0.987	0.832	0.077	0.036

For the data sets CLUS8, SPIR2, AGGR, SPIR3 and HORSE we find the interval values of Z, the only relevant parameter in our approach, maximizing NMI_{halo}. This interval is reported in Table 1 as $[Z_{best}]$, together with the corresponding value of NMI_{halo}. In the same table we also report the interval $[Z_{90\%}]$ within which the NMI_{halo} is at least 90% of its maximum value, and N_{clus} as the number of clusters inferred for Z in $[Z_{best}]$.

If the value of Z is appropriately chosen, the clustering algorithm is able to find all the ground-truth density peaks in all the considered artificial data sets with values of NMI_{dpa} close to 0.99, as shown in Table 1. However, the optimal choice for Z is not an isolated value, and the performance of the DPA clustering is not critically dependent on its exact choice: $[Z_{90\%}]$ show that already excellent results are obtained for a significantly large interval of Z values. The NMI estimated on the full set of points (NMI_{halo}) is also very large for most data sets. The worst performance is observed in SPIR3, where NMI_{halo} is equal to 0.809. This can be explained by a FPR_{halo} of 0.1 corresponding to a misclassification as 'halo' of 10% of data points that should instead belong to a cluster according to the ground-truth. A relatively large value of FPR_{halo} is observed also in the data set CLUS8. Indeed, the 'halo' by definition identifies all points whose assignment to a cluster is not reliable: this includes the 'noise' points, but also other points in low density regions. The values for FNR are instead very small, and even zero at times, indicating that the approach provides a reliable classification of the 'halo' points.

3.2. Clustering a handwritten digits data set

We further test our approach on the MNIST [24] data set, which includes 60,000 images of handwritten digits between 0 and 9. We compute the pairwise distances using the tangent distance [43], a metric explicitly developed for image comparison that is less sensible to transformations like rotation or translation. The intrinsic dimension of the data set estimated by the TWO-NN approach [12] is 8, and the value of Z is set to 1.6. The results are summarized in Fig. 4, with the topography description represented by the dendrogram in the top left panel and by the network at the bottom right. The color of each cluster in the topography representations derives from a majority rule assignation, namely the color is the one corresponding to the digit label with higher presence in the cluster whose color code is indicated under the matrix. The number of elements in each cluster is shown in italics between the dendrogram and the matrix that represents in a grey palette the fraction of points assigned to a cluster belonging to each of the ground truth labels: the darker the cell, the higher the fraction. As it can be seen from the matrix, the clusters contain almost always data points with a consistent ground truth classification. The number of clusters is larger than the ten classes of digits in the ground truth. For example, number seven is split in three clusters (9, 10, and 11) and number one in seven clusters. However, those belonging to the same digit appear as closely related to each other in the hierarchical structure represented by the dendrogram. The origin of this splitting can be in many cases ascribed to real differences in the handwritten digits: the same ground truth label is often assigned to images that look qualitatively different. Some examples are provided in the bottom left panel in Fig. 4, where the images belonging to cluster 24, 27, and 18 (number five), and those belonging to cluster 14, 15, and 16 (number four) look indeed qualitatively different. In the same panel we also show representative images of clusters 9, 10, and 11 (number seven) which look instead similar. Therefore, in this case the splitting of the three clusters is likely to be an artifact of our approach. With the aim of quantitatively measure the quality of the clustering result, we computed the NMI between the digit label assigned to each cluster according to a majority rule and the ground truth. We find a NMI of 0.84. A further analysis of this data set was done after undersampling it. In this case, only 10,000 out of 60,000 images were analyzed. The computed intrinsic dimension with TWO-NN [12] is equal to 7, which is coherent with the loss of information due to the undersampling, and the parameter Z is lowered to 1.0 to achieve a sufficient sensitivity to data structures, although accepting a possibly larger number of spurious clusters, as discussed in the Methods. The results are very similar to those obtained by analyzing the complete data set. The main difference is that the mixing between the digits four and nine in the reconstructed clusters is more significant, leading to a lower NMI of 0.76. The confusion matrices used for the NMI calculation for the full and the undersampled data set are respectively provided in Supp. Inf. Table S1 and S2.

3.3. The density topography of the PUA proteins clan

We finally exploit the approach introduced in this work to reconstruct the density topography of a sample of 9684 sequences extracted from the Pfam clan PUA. The Pfam database [14] is a large collection of protein families, grouped into clans or superfamilies; PUA is a complex superfamily organized into ten families with a population ranging from a few hundreds to thousands proteins, with many families containing a variety of protein architectures.

We first compute the local pairwise distances between the sequences by using a Modified Hamming distance described in Ref. [13]. The intrinsic dimension of the data set estimated by TWO-NN [12] is equal to 9. If one uses the standard k-NN density estimator to cluster the PUA protein sequences the choice of the optimal global k is far from trivial. The PAk density estimator, not surprisingly, finds a huge variability in the optimal values for k, ranging from 3 to 170 across the data set, reflecting the complexity of the sample. Using the PAk density estimator and Z equal to 2, we find 123 clusters.

We test our results against the Pfam classification of PUA sequences into families, and going into greater detail, into architectures, by computing the purity of our clusters. Here the purity of a cluster C with respect to an architecture A is defined as the number of sequences in C belonging to A divided by the total population of C. In Fig. 5 we represent the correspondence between clusters ordered according to the dendrogram (y-axis) and architectures (x-axis). Again the network representation
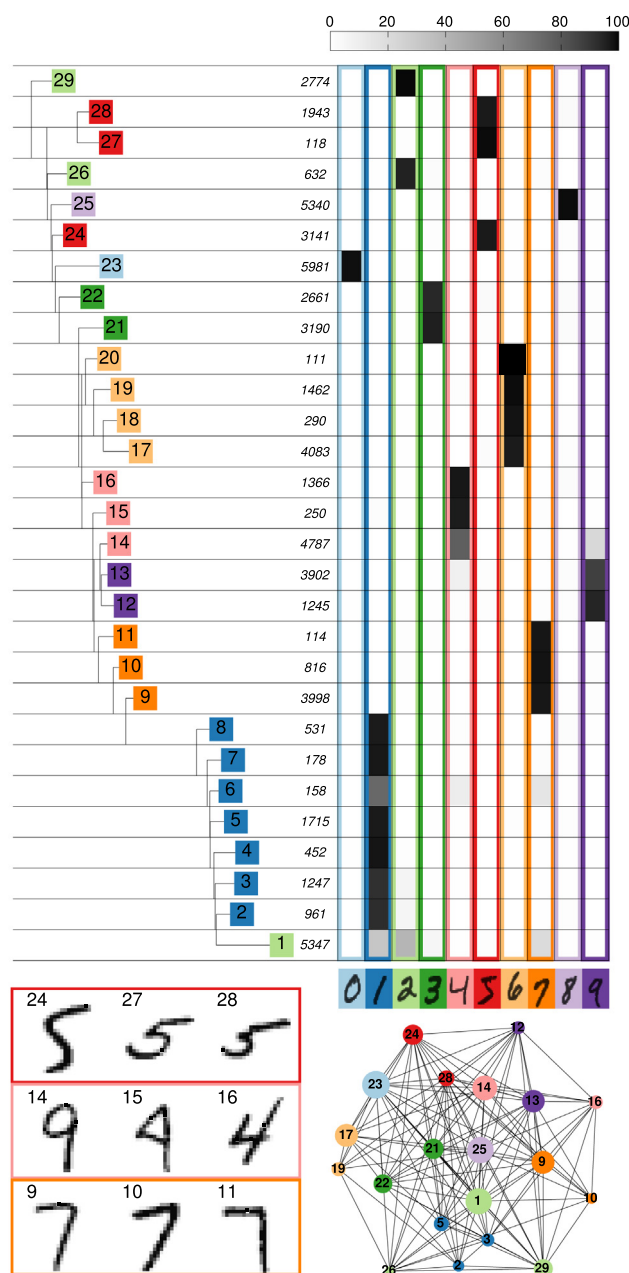
Fig. 4. Cluster analysis of the MNIST data set. In the matrix we represent, for clusters with a population greater than 100, the fraction of elements assigned to a cluster belonging to each of the ground truth labels. The higher the fraction, the darker the cell. On the left we show the clusters dendrogram. In correspondence to each leaf we indicate the cluster label and its population. To simplify the interpretation, the clusters background color is chosen according with a majority rule. For visualization purposes, only clusters with population higher or equal than 100 are shown. Bottom right: the clusters with population higher than 600 in a network representation. Bottom left: images corresponding to the centers of clusters assigned to numbers 5, 4 and 7.

is shown at the bottom of the graph. Only architectures and clusters with a population greater than 40 are displayed. The Pfam denomination of the architectures considered in Fig. 5 is provided in Supp. Inf. In this representation the purity of clusters with respect to architectures is associated to a grey palette: the darker the cell, the higher the purity. Fig. 5 shows that clusters are substantially pure with respect to architectures (most of the clusters are over 90% pure). The quality of the results was also assessed by computing the NMI [48] of the clustering partition with respect to the Pfam classification. Due to the hierarchical nature of the method, to compute the indices a family (or architecture) label is assigned to each cluster according to a majority rule. We find a NMI of 0.978 for the classification in families, and of 0.871 for the classification in
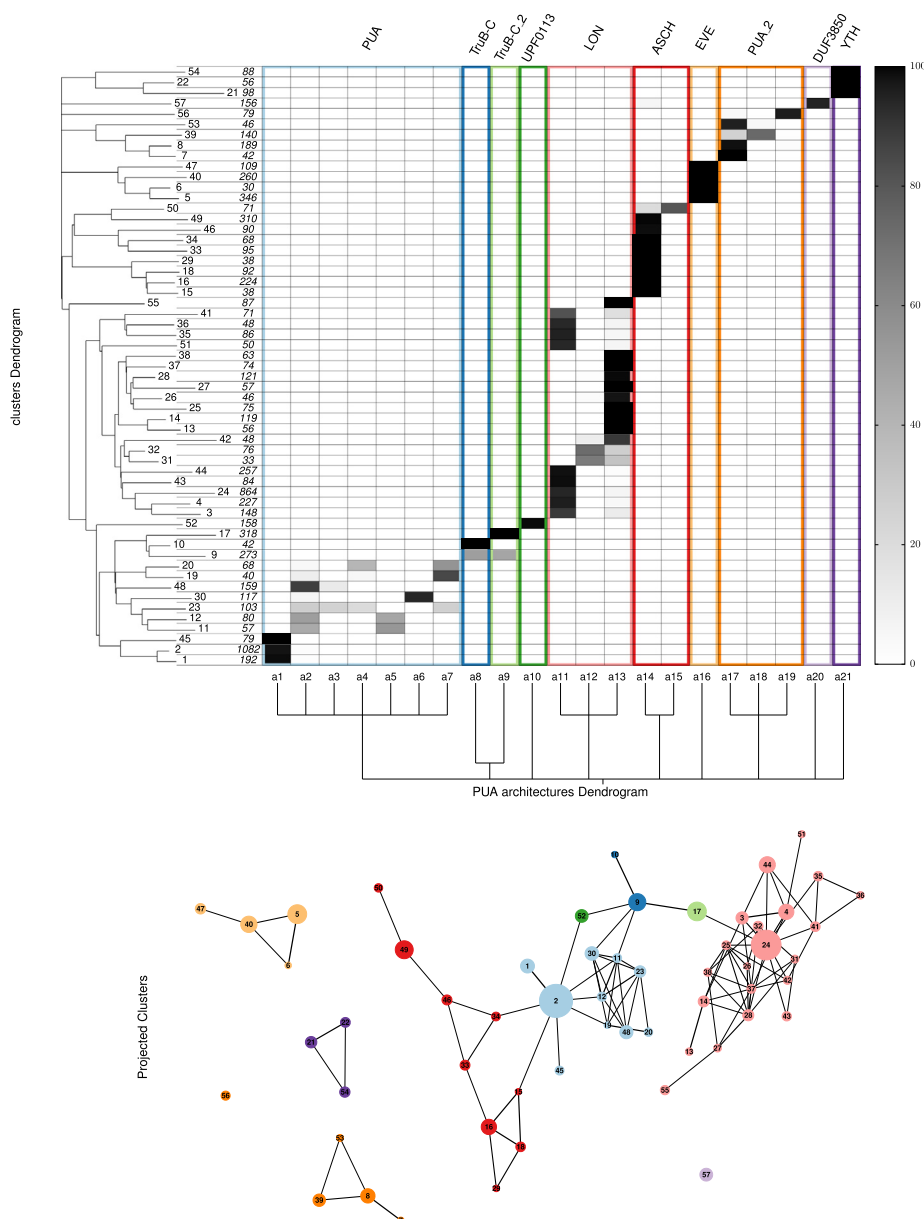
Fig. 5. Cluster analysis of the clan PUA from the Pfam database. We represent the Purity Matrix for clusters and architectures with a population greater than 40. Color boxes correspond to families. The purity of the clusters with respect to architectures is associated to a grey palette: the darker the cell, the higher the purity. On the left of the Purity Matrix we show the dendrogram of the clusters. In correspondence to each leaf we indicate the cluster label and the population of the cluster. The dendrogram at the bottom is a schematic visualization of the hierarchical relationship existing between architectures according to Pfam: architectures connected at a higher level (e.g. a1, a2, a3) belong to the same family, while those connected at a lower level (e.g. a12, a13) belong to a clan. The Pfam denomination of the architectures is provided in Supp. Inf. Bottom: network representation of the clusters represented in the top panel. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

architectures, which reveals a high degree of similarity between the clustering partition and the Pfam classification. The considerations that one should take into account when comparing with other methods (Fig. 6) are the same as in the case of MNIST. The dendrogram provides further information on the complex topography of the data set, showing, for instance, that clusters belonging to the same architecture are closely related to each other. It essentially reflects the similarity between families in the clan as well as their division into architectures. The only important exception is that cluster 9 is divided between families TruB-C_2 and TruB-C. These two families are characterized by sequences with a low similarity within the same family, thus the error in the estimated densities is so large that the faint saddle point that separates the two families is classified by our algorithm as a statistical fluctuation.
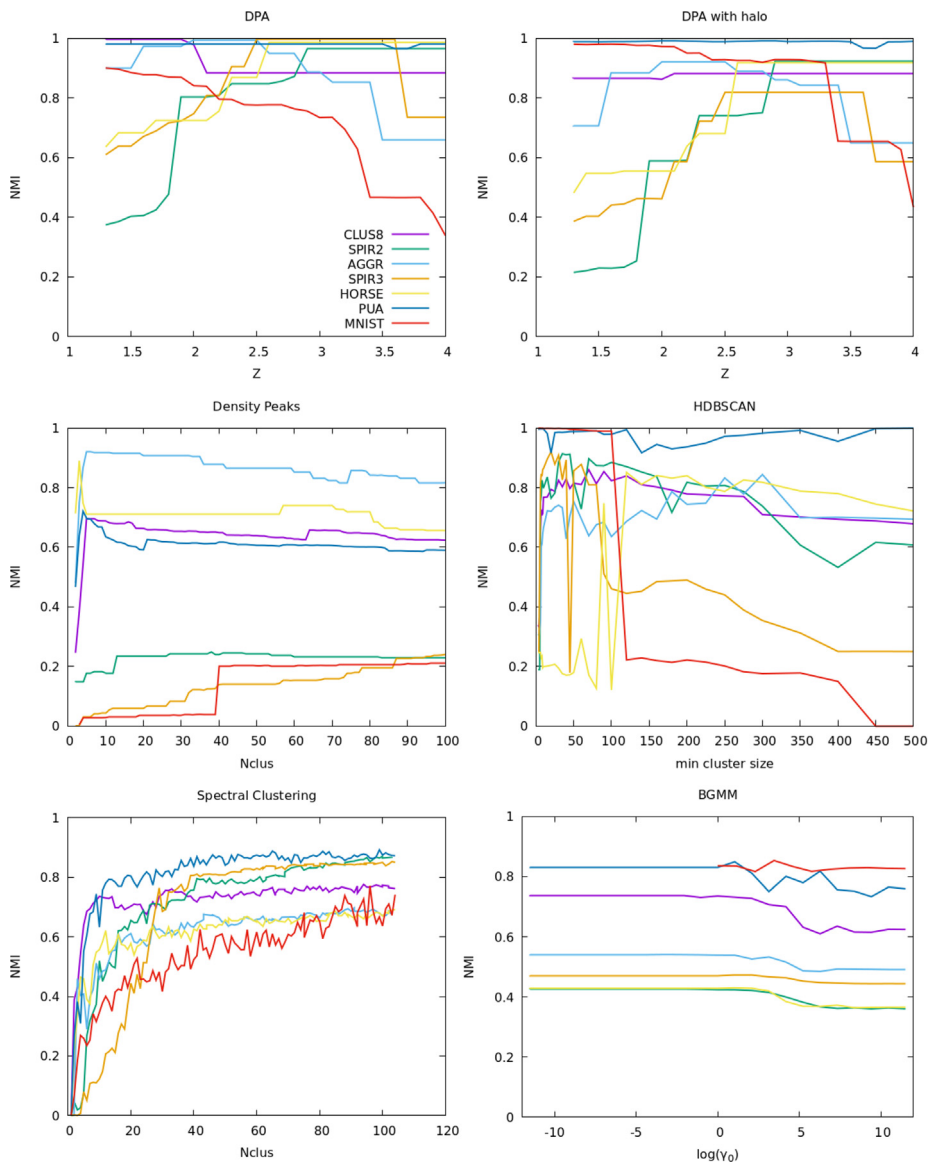
Fig. 6. Performance comparison between the new method, denoted as DPA (Density Peaks *Advanced*), and a selection of clustering methods. The influence of the algorithms parameters on the Normalized Mutual Information (NMI) [48] is evaluated using the artificial data sets CLUS8, SPIR2, AGG, SPIR3 and HORSE considered in in Table 1, and the real world data sets MNIST and PUA. For Spectral Clustering [32], Density Peaks [38] and the Gaussian Mixture Model [2] we compute the NMI by ignoring the points classified as 'noise' when present in the ground-truth. To evaluate the HDBSCAN [5] results on PUA and MNIST data sets instead we compute the NMI by ignoring the points labeled as 'noise' by the algorithm. Panel A: NMI as a function of Z, free parameter for the DPA clustering, computed by assigning all points to a cluster and ignoring those classified as 'noise' when present in the ground-truth. Panel B: NMI as a function of Z, free parameter for the DPA clustering, computed by including the 'halo' points. Panle C: NMI as a function of the number of clusters N_c, free parameter for DP clustering. Panel D: NMI as a function of the *min_cluster_size*, free parameter for HDBSCAN. Panel E: NMI as function of the number of clusters N_{cl}, free parameter for Spectral Clustering. Panel F: NMI as function of the *weight_concetration_parameter* γ_0, free parameter for Gaussian Mixture Model.

The network representation shows a complex landscape. For instance, while some families are well isolated others are interconnected through one or several nodes. The families PUA and LON, with the main cluster nodes 2 and 24 respectively, are divided in many clusters but they are densely interconnected between them. On the contrary, the family ASCH, although connected, appears to be quite sparse. The centrality of cluster 9 between families TruB-C_2 and TruB-C is in agreement with the analysis of the dendrogram.

3.4. Comparison with other methods

We compare our approach with other four state-of-the-art methods for clustering multidimensional data sets: Spectral Clustering [32], HDBSCAN [5], Gaussian Mixture Model [2], and DP clustering [38]. The comparison is performed using the artificial data sets CLUS8, SPIR2, AGG, SPIR3 and HORSE considered in Table 1, and the data sets MNIST and PUA introduced in the previous Sections. Data are first clustered by using the Python implementations available for the selected clustering methods: *BayesianGaussianMixture* and *SpectralClustering* implementations from the *scikit-learn* library [35], the *hdbscan* library [28], and our implementation of the DP clustering. The similarity between the clustering partitions and the ground-truth classification of the data sets are then evaluated using the NMI [48]. For the PUA and the MNIST data sets, before performing the NMI calculation, we match the clusters to reference populations using the majority rule: all points within each cluster are labeled as the ground-truth class of highest frequency in the cluster. This way we account for those true substructures not classified by the ground-truth that are indeed reconstructed as different clusters.

In order to compare all the clustering methods on equal footing, we have to consider that Spectral Clustering, Density Peaks and the Gaussian Mixture Model do not attempt to recognize noise data points, while HDBSCAN assigns always a noise category. In the ground-truth of the artificial data sets some points are classified as 'noise', while no noise is expected in the PUA and MNIST data sets. Therefore we apply method specific calculations of the NMI when considering the artificial data sets or the real world ones. In particular for Spectral Clustering, Density Peaks and the Gaussian Mixture Model we compute the NMI by ignoring the points classified as 'noise' when present in the ground-truth. To evaluate the HDBSCAN results on PUA and MNIST data sets instead we compute the NMI by ignoring the points labeled as noise by the algorithm. In the case of DPA, we can compute the NMI by either assigning all points to a cluster, and ignoring the points classified as 'noise' when present in the ground-truth as done for Spectral Clustering, Density Peaks and the Gaussian Mixture Model, or considering the points classified as 'halo' in analogy to HDBSCAN.

The accuracy of each considered clustering algorithm is evaluated quantifying the influence of the algorithms parameters on the NMI. This is done by varying a single parameter of an algorithm while keeping the others at their default values. For the DPA clustering we compute the NMI as a function of the Z parameter. For Spectral Clustering, the free parameter is the total number of clusters N_{cl}. For the Bayesian Gaussian Mixture Model, we scan the *weight_concentration_prior* parameter for the Dirichlet process as prior type. For this method, the clustering of the high-dimensional data sets PUA and MNIST is preceded by a projection in a 2D space obtained using the t-SNE [27] dimensionality reduction (these projections, that are informative by themselves, are provided in Fig. S7 of Supp. Inf.). For the HDBSCAN method, we compute the NMI as a function of the *min_cluster_size* parameter, which fixes the minimum size of a cluster, while we set the parameter *min_samples* equal to it, as default option. In the standard DP method, we compute the NMI as a function of the number of clusters, with the peaks chosen in order of decreasing $\gamma = \rho \cdot \delta$. Following the original algorithm [38], the δ is defined as the distance of a data point from its nearest neighbour of higher density, and the density is estimated by the exponential kernel estimator whose width d_c is chosen in such a way that the average number of neighbors is the 2% of the total number of points in the data set.

The results are summarized in Fig. 6. For the DPA clustering we show the values of NMI computed by assigning all points to a cluster and ignoring those classified as 'noise' when present in the ground-truth in Panel A, and the NMI computed by including the 'halo' points in Panel B. The interval of Z values maximizing the NMI is data set dependent, however by setting $Z = 3$ the value of the NMI is larger than 0.8 for all the data sets in both Panel A and B, except for MNIST whose NMI is equal to 0.7 in Panel A. We also notice that in the high-dimensional data sets (MNIST and PUA) the best results are obtained with relatively low values of Z: due to the curse of dimensionality the typical error on the density estimate is large, which makes hard to distinguish between real clusters and statistical fluctuations. In Panel C we plot the NMI values for the DP clustering. The performance of this algorithm is poor, especially for the data sets MNIST, SPIR2 and SPIR3. This is due to the complexity of the density distribution in these data sets, which makes it difficult to identify the true clusters centers by selecting the peaks with higher γ, because the δ_i parameter is typically very small also for the cluster centers. In Panel D we report the value of NMI for HDBSCAN. In data sets SPIR3, HORSE and MNIST, the NMI does not vary smoothly as a function of the *min_cluster_size* parameter. The best results are obtained by choosing a small value of this parameter, but for all the possible choices there are at least two data sets for which the NMI is significantly smaller than 0.8. In Panel E we report the value of NMI for the Spectral Clustering approach. In this case the best performance is obtained by setting a high value for the free parameter, the number of clusters N_c. The NMI does not decrease when N_c becomes large because the new clusters are either small or in the points that are assigned as noise in the ground truth. The best values of the NMI are in many cases worse than those obtained with DPA, in particular for the datasets HORSE, AGG and CLUS8 the best NMI is lower than 0.8. Finally, in Panel F we plot the value of NMI for the Gaussian Mixture Model: the NMI is remarkably stable with respect to the free parameter, but the performance is in general worse than for the other methods we considered.

3.5. Computational cost

In order to check the run-time performance and scaling of the DPA clustering algorithm, we generated artificial data sets with a distribution defined by the mixture of two Gaussian and background noise, in dimensions (d) 2, 4, and 8, and with the number of data points (N) ranging from 2000 to 250,000. Therefore, 60 data sets with different combinations of $[N, d]$ were generated in total. For these tests, the time is defined as the total wall-clock time from start to finish, as measured by the *time*
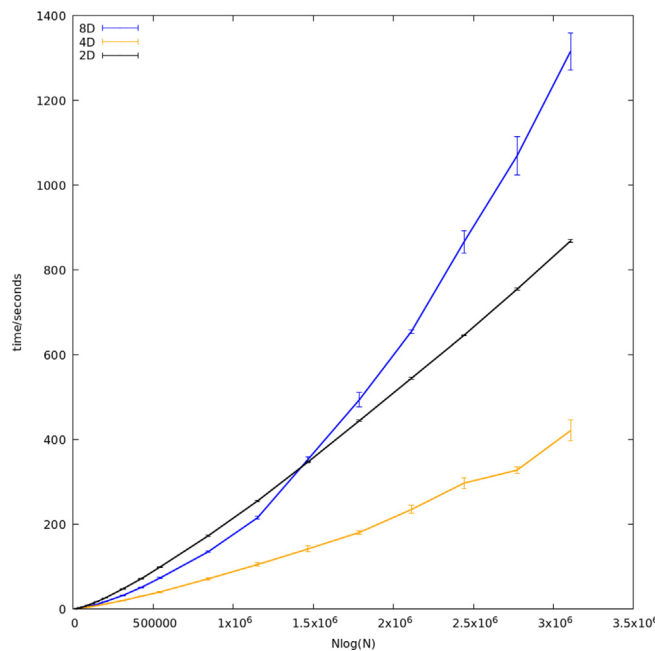
Fig. 7. Run times for the current implementation of the DPA algorithm for different number of data points N and embedding dimensions d. The test cases have been generated from 400,000 points that belong to a bimodal Gaussian distribution and then added 100,000 points of uniform background noise. Then, they have been undersampled from in a range that goes from 2000 to 250,000 points. The procedure has been repeated ten times for obtaining an average time and a standard deviation (error bars). In order to check the scaling with the dimension of the data set, the calculation has been done in dimension 2, 4 and 8. While at low number of samples, the behavior is complex, when they increase it can be seen that the time increases linearly with $N \cdot \log(N)$. The times showed correspond to calculations made in a Intel(R) Core(TM) i7-7800X CPU @ 3.50 GHz on Linux-5.3.0–19-generic-x86_64-with-debian-buster-sid.

$()$ function of the Python *time* module, and for each pair $[N, d]$ the average time and standard deviation are estimated over ten further realization of the data set. The parameter Z has been set to a reasonable large value ($Z = 5$) in order to provide an upper bound estimation of the time: as defined by Heuristic 3, the number of merging steps increases with the value of Z, and therefore also the run time of the algorithm.

Results are summarized in Fig. 7, where we plot the average time as a function of $N \cdot \log(N)$. It can be seen that the clustering time for 250,000 points is about 15, 5, and 20 min respectively for d equals to 2, 4, and 8, so at relatively high dimensions the run time is of the order of a fraction of an hour. The aforementioned times are obtained on an Intel(R) Core(TM) i7-7800X CPU @3.50 GHz using the Python implementation of DPA (see Code availability Section). This implementation heavily relies on the Scikit-learn [35] Nearest Neighbors search, currently implemented with library that uses a heuristic selecting between brute-force, ball-tree [33] and kd-tree [1] algorithms. The scaling of the algorithm for large values of N is $N \cdot \log(N)$, as shown by the trends seen in Fig. 7. The scaling with the dimensionality d is more complex, because an increased value for d usually corresponds to a reduction in the optimal neighborhood size $\hat{k}$ around each point. Therefore, while the computational cost of the Nearest Neighbor search increases with d, the steps in the algorithm depending on the value of $\hat{k}$ show a better performance. This explains why the run times estimated at $d = 4$ are lower than those at $d = 2$. However, this behavior depends on the specific characteristics of the data set, and it is hard to generalize.

4. Conclusion

In this work we introduce a tool to analyze large and multidimensional data sets, which is specifically designed to treat cases in which the standard projection technique (e.g. PCA) gives poor results.

Data sets can often be described as realizations of an underlying probability distribution, whose density has support in the space of the features (coordinates) of the data. Therefore, the idea is to reconstruct the *topography* of the probability distribution from which the data are generated. Our clusters topography is a list of probability peaks, each characterized by its properties: the height of the probability maximum, the population, the list of neighboring peaks, etc. Our tool employs only the distance between points, avoiding the use of any coordinate system or projection, and its derivation is based on three assumptions: (i) the data points are generated independently from a probability distribution function; (ii) this probability distribution is continuous and (iii) the distance between data points is a metric, and in particular it should satisfy the triangle inequality.

We illustrated two ways to visualize this topography: a hierarchical representation equivalent to the one used in Refs. [19,5,8,44] and a graph representation tantamount to the one of Markov State Models [34]. Of course, one can imagine other graphical representations.

The topography is reconstructed by a modified version of the unsupervised Density Peaks clustering algorithm [38] whose key ingredient is the PAk density estimator. The PAk estimator is parameter free and, with respect to other non-parametric estimators, provides an accurate estimation of the probability density at the data points in the manifold in which they lay, and not in the space of coordinates. It also provides the error associated with the density estimate: this is crucial to assess the statistical significance of the peaks found by the clustering procedure and, as a consequence, to discriminate between real features of the underlying probability distribution function and artifacts due to finite sampling. The use of PAk coupled with the Density Peaks clustering algorithm thus leads to an accurate detection of the main features of the underlying probability distribution, where the statistical reliability of the probability peaks is quantified by the Z score measuring the negative of the probability that the peak is generated by a statistical fluctuation. By choosing a threshold on Z, one can filter out the peaks that are less reliable and obtain a more compact representation of the data. A drawback of the method is that, due to the curse of dimensionality, the PAk algorithm does not work well when the intrinsic dimension of the data set is higher than 10–20 [37], and therefore neither will DPA. However, many important data sets lay in a manifold that can be twisted and topologically complex but whose intrinsic dimension is typically much smaller than the number of coordinates of the system [25], which is instead typically very large. The method performs well in the two dimensional toy examples in Figs. 1 and 3. The artificial low-dimensional data sets AGG, SPIR3 and HORSE are obtained respectively from the Aggregation [16], Spirals [7] and Jain [22] data set, well-known as test sets for investigating density-based clustering algorithms. In Ref. [23] for example Aggregation, Spirals and Jain are used to test the performance of the algorithms on irregular-shaped clusters or clusters of varying sizes. However, real-world data sets are usually noisy, and background noise is not present in those data sets. Moreover, if optimized on data sets with well-separated clusters, algorithms may be less performant on real-world data sets where clusters are typically connected by regions with a lower density of data points, as in the case of Ref. [49]. Therefore, for our experiments we generated AGG, SPIR3, and HORSE using a probability distribution function built as a sum of Gaussian functions centered in the original data points. The resulting data sets maintain the same number of clusters and dimensionality of the original ones, with the addition of noise points to better resemble the continuous distribution of data points typical of real-world data sets. Furthermore, we investigated the method on handwritten numbers (NMIST) and the PUA clan with intrinsic dimension of 8 and 9 respectively, as estimated by the TWO-NN method [12]. The MNIST handwritten digits data set is largely used in Machine Learning to benchmark density-based clustering methods, although the presence of different handwritings style and the high dimensionality of the data set can make the correct identification of ciphers challenging, as shown in Ref. [21]. The PUA data set has been recently analyzed also in Ref. [40] for automatic protein domain classification.

The method shows its real power when the number of features is huge, as in the cases of MNIST and the PUA clan where computing the density in the manifold in which the data lay, instead of computing it in the coordinate space, is the key for a successful reconstruction of the topography. Moreover, the knowledge of the density at the borders permits the visualization of the relationship between these modes – the topography– in several ways providing a visual grasp of the structure of the data set with an unprecedented level of detail.

Code availability

A Python implementation of the algorithm is available on github, at https://github.com/mariaderrico/DPA. It is also available a Fortran implementation at: https://github.com/alexdepremia/Advanced-Density-Peaks.

CRediT authorship contribution statement

Maria d'Errico: Conceptualization, Methodology, Investigation, Software, Validation, Formal analysis, Visualization, Writing - original draft, Writing - review & editing. **Elena Facco:** Data curation, Formal analysis, Visualization, Writing - original draft, Writing - review & editing. **Alessandro Laio:** Conceptualization, Methodology, Investigation, Formal analysis, Visualization, Writing - original draft, Writing - review & editing. **Alex Rodriguez:** Conceptualization, Methodology, Investigation, Software, Validation, Formal analysis, Visualization, Writing - original draft, Writing - review & editing.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

Appendix A. Supplementary data

Supplementary data associated with this article can be found, in the online version, at https://doi.org/10.1016/j.ins.2021.01.010.

References

[1] J.L. Bentley, Multidimensional binary search trees used for associative searching, Commun. ACM 18 (1975) 509–517.
[2] D.M. Blei, M.I. Jordan, et al, Variational inference for dirichlet process mixtures, Bayesian Anal. 1 (2006) 121–143.
[3] K. Bunte, M. Biehl, B. Hammer, A general framework for dimensionality-reducing data visualization mapping, Neural Comput. 24 (2012) 771–804.
[4] F. Camastra, A. Staiano, Intrinsic dimension estimation: advances and open problems, Inf. Sci. 328 (2016) 26–41.
[5] R.J. Campello, D. Moulavi, J. Sander, Density-based clustering based on hierarchical density estimates, in: Pacific-Asia Conference on Knowledge Discovery and Data Mining, 2013, Springer, pp. 160–172.
[6] M. Ceriotti, G.A. Tribello, M. Parrinello, Simplifying the representation of complex free-energy landscapes using sketch-map, Proc. Nat. Acad. Sci. 108 (2011) 13023–13028.
[7] H. Chang, D.Y. Yeung, Robust path-based spectral clustering, Pattern Recogn. 41 (2008) 191–203.
[8] K. Chaudhuri, S. Dasgupta, S. Kpotufe, U. von Luxburg, Consistent procedures for cluster tree estimation and pruning, IEEE Trans. Inf. Theory 60 (2014) 7900–7912.
[9] R.R. Coifman, S. Lafon, A.B. Lee, M. Maggioni, B. Nadler, F. Warner, S.W. Zucker, Geometric diffusions as a tool for harmonic analysis and structure definition of data: diffusion maps, Proce. Nat. Acad. Sci. USA 102 (2005) 7426–7431.
[10] D. Comaniciu, P. Meer, Mean shift: a robust approach toward feature space analysis, IEEE Trans. Pattern Anal. Mach. Intell. 24 (2002) 603–619.
[11] M. Ester, H.P. Kriegel, J. Sander, X. Xu, A density-based algorithm for discovering clusters in large spatial databases with noise, in: Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, AAAI Press, 1996, pp. 226–231.
[12] E. Facco, M. d'Errico, A. Rodriguez, A. Laio, Estimating the intrinsic dimension of datasets by a minimal neighborhood information, Sci. Rep. 7 (2017) 12140.
[13] E. Facco, A. Pagnani, E.T. Russo, A. Laio, The intrinsic dimension of protein sequence evolution, PLOS Comput. Biol. 15 (2019) 1–16.
[14] R.D. Finn, A. Bateman, J. Clements, P. Coggill, R.Y. Eberhardt, S.R. Eddy, A. Heger, K. Hetherington, L. Holm, J. Mistry, E.L.L. Sonnhammer, J. Tate, M. Punta, Pfam: the protein families database, Nucleic Acids Res. 42 (2014) D222–D230.
[15] R.D. Finn, P. Coggill, R.Y. Eberhardt, S.R. Eddy, J. Mistry, A.L. Mitchell, S.C. Potter, M. Punta, M. Qureshi, A. Sangrador-Vegas, G.A. Salazar, J. Tate, A. Bateman, The Pfam protein families database: towards a more sustainable future, Nucleic Acids Res. 44 (2016) D279–D285.
[16] A. Gionis, H. Mannila, P. Tsaparas, Clustering aggregation, ACM Trans. Knowl. Discovery Data (TKDD) 1 (2007) 4.
[17] A. Gisbrecht, B. Hammer, Data visualization by nonlinear dimensionality reduction, Wiley Interdisc. Rev. Data Min. Knowl. Discovery 5 (2015) 51–73.
[18] D. Granata, V. Carnevale, Accurate estimation of the intrinsic dimension using graph distances: Unraveling the geometric complexity of datasets, Sci. Rep. 6 (2016) 31377.
[19] J.A. Hartigan, Consistency of single linkage for high-density clusters, J. Am. Stat. Assoc. 76 (1981) 388–394.
[20] J.A. Hartigan, Consistency of single linkage for high-density clusters, J. Am. Stat. Assoc. 76 (1981) 388–394.
[21] S. Hess, W. Duivesteijn, P. Honysz, K. Morik, The spectacl of nonconvex clustering: a spectral approach to density-based clustering, in, in: Proceedings of the AAAI Conference on Artificial Intelligence, 2019, pp. 3788–3795.
[22] A.K. Jain, M.H. Law, Data clustering: a user's dilemma, in: International Conference on Pattern Recognition and Machine Intelligence, 2005, Springer, pp. 1–10.
[23] J. Jiang, Y. Chen, X. Meng, L. Wang, K. Li, A novel density peaks clustering algorithm based on k nearest neighbors for improving assignment process, Physica A 523 (2019) 702–713.
[24] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, Proc. IEEE 86 (1998) 2278–2324.
[25] E. Levina, P.J. Bickel, Maximum likelihood estimation of intrinsic dimension, in: L.K. Saul, Y. Weiss, L. Bottou (Eds.), Advances in Neural Information Processing Systems 17, MIT Press, 2005, pp. 777–784.
[26] Z. Liang, P. Chen, Delta-density based clustering with a divide-and-conquer strategy: 3dc clustering, Pattern Recogn. Lett. 73 (2016) 52–59.
[27] L.v.d. Maaten, G. Hinton, Visualizing data using t-sne, J. Mach. Learn. Res. 9 (2008) 2579–2605.
[28] L. McInnes, J. Healy, S. Astels, hdbscan: hierarchical density based clustering, J. Open Source Software 2 (2017).
[29] R. Mehmood, G. Zhang, R. Bie, H. Dawood, H. Ahmad, Clustering by fast search and find of density peaks via heat diffusion. Neurocomputing 208 (2016) 210–217. SI: BridgingSemantic.
[30] M.C. Minnotte, Nonparametric testing of the existence of modes, Ann. Stat. (1997) 1646–1660.
[31] J. Neyman, E.S. Pearson, On the problem of the most efficient tests of statistical hypotheses, Philos. Trans. Roy. Soc. Lond. Ser. A Contain. Papers Math. Phys. Charact. 231 (1933) 289–337.
[32] A.Y. Ng, M.I. Jordan, Y. Weiss, On spectral clustering: analysis and an algorithm, Adv. Neural Inf. Process. Syst. (2002) 849–856.
[33] S.M. Omohundro, Five Balltree Construction Algorithms, International Computer Science Institute Berkeley, 1989.
[34] V.S. Pande, K. Beauchamp, G.R. Bowman, Everything you wanted to know about markov state models but were afraid to ask. Methods 52 (2010) 99–105. Protein Folding.
[35] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, E. Duchesnay, Scikit-learn: machine learning in Python, J. Mach. Learn. Res. 12 (2011) 2825–2830.
[36] M. Ringnér, What is principal component analysis?, Nat. Biotechnol. 26 (2008) 303–304.
[37] A. Rodriguez, M. d'Errico, E. Facco, A. Laio, Computing the free energy without collective variables, J. Chem. Theory Comput. 14 (2018) 1206–1215.
[38] A. Rodriguez, A. Laio, Clustering by fast search and find of density peaks, Science 344 (2014) 1492–1496.
[39] S.T. Roweis, L.K. Saul, Nonlinear dimensionality reduction by locally linear embedding, Science 290 (2000) 2323–2326.
[40] E.T. Russo, A. Laio, M. Punta, Dpcfam: a new method for unsupervised protein family classification, 2020. bioRxiv.
[41] A.D. Shieh, T.B. Hashimoto, E.M. Airoldi, Tree preserving embedding, Proc. Nat. Acad. Sci. 108 (2011) 16916–16921.
[42] B.W. Silverman, Using kernel density estimates to investigate multimodality, J. Roy. Stat. Soc. Ser. B (Methodol.) (1981) 97–99.
[43] P. Simard, Y. LeCun, J.S. Denker, Efficient pattern recognition using a new transformation distance, Adv. Neural Inf. Process. Syst. (1993) 50–58.
[44] F. Sittel, G. Stock, Robust density-based clustering to identify metastable conformational states of proteins, J. Chem. Theory Comput. 12 (2016) 2426–2435.
[45] G. Sormani, A. Rodriguez, A. Laio, Explicit characterization of the free-energy landscape of a protein in the space of all its cα carbons, J. Chem. Theory Comput. 16 (2019) 80–87.
[46] J.B. Tenenbaum, V. De Silva, J.C. Langford, A global geometric framework for nonlinear dimensionality reduction, Science 290 (2000) 2319–2323.
[47] W.S. Torgerson, Multidimensional scaling: I. Theory and method, Psychometrika 17 (1952) 401–419.
[48] N.X. Vinh, J. Epps, J. Bailey, Information theoretic measures for clusterings comparison: variants, properties, normalization and correction for chance, J. Mach. Learn. Res. 11 (2010) 2837–2854.
[49] Y. Wang, W. Pang, Y. Zhou, Density propagation based adaptive multi-density clustering algorithm, Plos One 13 (2018) e0198948.
[50] D. Xu, Y. Tian, A comprehensive survey of clustering algorithms, Ann. Data Sci. 2 (2015) 165–193.