Determinantal consensus clustering

Serge Vicente · Alejandro Murua-Sazo

Received: 21 January 2021 / Revised: 16 June 2022 / Accepted: 27 July 2022 / Published online: 25 August 2022
© Springer-Verlag GmbH Germany, part of Springer Nature 2022

Abstract
Random restart of a given algorithm produces many partitions that can be aggregated to yield a consensus clustering. Ensemble methods have been recognized as more robust approaches for data clustering than single clustering algorithms. We propose the use of determinantal point processes or DPPs for the random restart of clustering algorithms based on initial sets of center points, such as $k$-medoids or $k$-means. The relation between DPPs and kernel-based methods makes DPPs suitable to describe and quantify similarity between objects. DPPs favor diversity of the center points in initial sets, so that sets with similar points have less chance of being generated than sets with very distinct points. Most current initial sets are generated with center points sampled uniformly at random. We show through extensive simulations that, contrary to DPPs, this technique fails both to ensure diversity, and to obtain a good coverage of all data facets. The latter are two key properties that make DPPs achieve good performance. Simulations with artificial datasets and applications to real datasets show that determinantal consensus clustering outperforms consensus clusterings which are based on uniform random sampling of center points.

Keywords Classification · Kernel-based validation index · Kernel $k$-means · Partitioning about medoids · Radial basis function · Repulsion · Voronoi diagram

Mathematics Subject Classification 62H30 · 62-08

1 Introduction
A classical procedure in fields such as biology, psychology, medicine and marketing is to group elements based on similar features (cluster analysis) to provide a framework...
for learning (Jain and Dubes 1988). Some clustering techniques, such as the standard $k$-means algorithm (Lloyd 1982) or the partitioning around medoids algorithm (Kaufmann and Rousseeuw 1987), are characterized by an initial choice of a subset of random points. We find the same type of initial choice in some classification techniques, such as neural networks or machine learning models that use the stochastic gradient descent algorithm for optimization (Saad 1998). Selecting a subset of points simply at random does not consider the diversity among the selected points, because this sort of sampling mechanism gives to every point an equal probability of being selected. Many similar points may be chosen simultaneously, conveying much redundancy and little representability of the data. In some domains of research, the diversity of the selected points is a major concern. Ensuring diversity, so as to obtain a good coverage of all data facets, would certainly fail when doing simple random sampling. In contrast, determinantal point processes, or DPPs for short, induce negative correlations between similar points (Borodin and Olshanski 2000). They permit to assign higher probability to sets of points that are more diverse (Kulesza and Taskar 2012). Consequently, similar points have less chance of appearing together. This property has established DPPs’ use in machine learning as models for subset selection (Hafiz Affandi et al. 2013).

Traditionally, cluster analysis methods are gathered in two families: hierarchical and partitioning non-hierarchical techniques. We consider only the so-called crisp versions of the methods, where each data point is assigned to a single cluster (Lago-Fernández and Corbacho 2010). Hierarchical techniques include agglomerative clustering with single linkage (Florek et al. 1951) and Ward linkage (Ward Jr 1963), and Hidden Markov Models agglomerative clustering (Smyth 1997). Partitioning techniques include the partitioning around medoids (PAM) algorithm (Kaufmann and Rousseeuw 1987), and the $k$-means algorithm (Lloyd 1982). Other techniques have emerged later, such as probabilistic model-based techniques, (Banfield and Raftery 1993; Fraley and Raftery 1998), density-based techniques (Ester et al. 1996; Ankerst et al. 1999; Stuetzle and Nugent 2010; Stuetzle 2003), and grid-based techniques (Wang et al. 1997; Hinneburg and Keim 1999). One can find a good description of some of these techniques in (Han et al. 2011). Recently due to the contribution of advanced computational methods, additional families of clustering have arised. Among these, spectral clustering (Shi and Malik 2000; Ng et al. 2001) has gained popularity. It extracts the eigenvalues of a similarity matrix to perform dimensionality reduction before doing clustering. This technique can also be framed within the context of kernel based clustering methods, and hence it is linked to many other similar approaches (Filippone et al. 2008).

Most clustering methods seek to obtain a single optimal partition of the data according to some internal clustering criterion based on the principle of maximizing both within-cluster similarity and between-cluster dissimilarity. Different clustering techniques applied to the same data can produce very different clustering results, due in part to a lack of an external objective and impartial criterion (Vega-Pons and Ruiz-Shulcloper 2011). For some techniques their dependency on the initial choice of center points (centroids or medoids) can also explain those differences. In order to improve the quality and robustness of clustering results Blatt et al. (1996, 1997) introduced a probabilistic cluster membership framework. Strehl and Ghosh (2002) formalized this
Determinantal consensus clustering approach by defining the *cluster ensembles* approach whose main objective is to combine different clustering results into a single consolidated clustering. Vega-Pons and Ruiz-Shulcloper (2011) established four desirable properties that should be present in any cluster ensemble method: robustness, consistency, novelty and stability. In particular, these imply that any cluster ensemble method should (i) produce clustering solutions usually not attainable by single clustering algorithm, and (ii) have lower sensitivity to noise, outliers and initial conditions. One of the most well-known cluster ensembles methods was introduced by Monti et al. (2003) in genomic studies, inspired by resampling and cross-validation techniques such as bootstrapping.

More generally, the cluster ensemble produces multiple partitions generated by possibly different clustering algorithms. The multiple partitions may also be obtained by multiple executions of the same clustering algorithm. Consensus clustering is defined as a method meant for attaining a single consolidated clustering from the multiple partitions. The obtained single consolidated clustering, built over some agreement among the several runs, represents a partition of data. Although not required, multiple runs of the algorithm could be initialized with a random restart.

In this paper, we focus on partitioning techniques with random initial conditions. We explore the determinantal point process for sampling the initial cluster centers (Ben Hough et al. 2006; Kulesza and Taskar 2012). It involves the choice of a real, symmetric and positive semidefinite matrix that measures similarity between each pair of points. The properties of this type of matrices open a connection with the well-known kernel-based methods, which have been widely used in pattern analysis, classification and clustering (Howley and Madden 2006). One of the most popular kernels, and the one we use in this paper, is the Radial Basis Function, also known as the Gaussian kernel.

The paper is organized as follows: in Sect. 2, we present the basic ideas related to consensus clustering. In Sect. 3, we lay out the basic properties of determinantal point processes, and put them into context as a sampling method to generate cluster centers in a partitioning clustering algorithm. Since the choice of a proper kernel is central in the construction of determinantal point processes, kernel-based methods are briefly described in this section as well. In Sect. 4, we introduce our proposed methodology for determinantal consensus clustering, or *consensus DPP*, for short. In Sect. 5, we perform an extensive simulation in order to evaluate the performance of consensus DPP. The performance of the proposed algorithm on real datasets is shown in Sect. 6. Comparisons with other partitioning methods are also shown in these last two sections. We conclude with a few thoughts and a discussion in Sect. 7.

## 2 Consensus clustering

Throughout the paper the data will be denoted by $S = \{x_1, \ldots, x_n\} \subset \mathbb{R}^p$, where $x_i$ represents a $p$-dimensional vector, for $i = 1, \ldots, n$ and $n \geq 2$. As mentioned in Sect. 1, this paper focuses on *crisp* partitions of the data. Consider a particular partitioning clustering technique run $R$ times on the data $S$. The agreement among the several runs of the algorithm is based on the *consensus matrix* $C$. This is a $n \times n$ symmetric matrix whose entries $\{C_{ij}, i, j = 1, \ldots, n\}$ represent the proportion of runs
in which elements \( x_i \) and \( x_j \) of \( S \) fall in the same cluster. Let \( r \) represent a specific run of the clustering algorithm, \( r = 1, \ldots, R \), and let \( C_r = (c_{ij}^r) \) be the associated \( n \times n \) symmetric binary matrix with entries given by \( c_{ij}^r = 1 \) if \( x_i \) and \( x_j \) belong to the same cluster, and \( c_{ij}^r = 0 \), otherwise, \( i, j = 1, \ldots, n \). The consensus matrix \( C \) is then the \( n \times n \) symmetric matrix with entries defined by \( C_{ij} = \sum_{r=1}^{R} c_{ij}^r / R \), \( i, j = 1, \ldots, n \). The entry \( C_{ij} \) is known as consensus index. Obviously, the diagonal entries are given by \( C_{ii} = 1 \), for \( i = 1, \ldots, n \). Clusters can also be defined using graph theory by considering the graph \((V, E)\) whose vertices are given by the elements \( x_i \in S \). The set of edges \( E \) is defined by connecting each pair of elements sharing some common features or properties. A clustering configuration with \( K \) clusters consists in an undirected graph with \( K \) connected components. The single consolidated clustering with \( K \) final clusters obtained by consensus clustering are the \( K \) connected components of the consensus graph (Murua et al. 2008), that is, the graph over the observations with an edge between any pair of elements that belong to the same cluster in the majority of the configurations. The majority concept is tied to the consensus index defined above.

### 3 The determinantal point process

Keeping the notation from the previous section, a DPP is a probability measure on \( 2^S \) that assigns the probability

\[
P(Y) = \frac{\det(L_Y)}{\det(L + I_n)},
\]

(1)

to any subset \( Y \in 2^S \), where \( L \) is a \( n \times n \) real, symmetric and positive semidefinite matrix that measures similarity between all pairs of elements of \( S \); \( L_Y \) is the principal submatrix of \( L \) whose rows and columns are indexed by \( Y \), i.e., \( L_Y = (L_{ij})_{i,j \in Y} \); and \( I_n \) is the \( n \times n \) identity matrix. If \( Y \) is the random variable that represents the subset selected from \( 2^S \), then we write \( Y \sim DPP_{S}(L) \) for the corresponding determinantal process. The matrix \( L \) is known as the kernel matrix of the (Kulesza and Taskar 2012; Kang 2013; Hafiz Affandi et al. 2014). It can be shown (Kulesza and Taskar 2012) that \( \det(L + I_n) = \sum_{Y \subset 2^S} \det(L_Y) \), hence (1) does indeed define a probability mass function over all subsets in \( 2^S \). This definition states restrictions on all the principal minors of the kernel matrix \( L \), denoted by \( \det(L_{Y}) \). Indeed, as \( P(Y = Y) \propto \det(L_Y) \) represents a probability measure, we have \( \det(L_{Y}) \geq 0 \), for any \( Y \subseteq S \). This implies that any symmetric positive semidefinite matrix can be taken as kernel matrix \( L \), where its eigenvalues are such that \( \lambda_i(L) \geq 0 \), for every \( i = 1, \ldots, n \).

Determinants have a well-known geometric interpretation. Because \( L \) is positive semidefinite, it can be decomposed as \( L = B^T B \), where \( B \) is a \( m \times n \) matrix. Denoting the columns of \( B \) by \( B_i \), for \( i = 1, \ldots, n \), we have

\[
P(Y = Y) \propto \det(L_Y) = \text{Vol}^2 (\{B_i\}_{i \in Y}),
\]

(2)

where \( \text{Vol}^2 \) represents the squared volume of the parallelepiped spanned by the columns of \( B \) corresponding to elements in \( Y \). The columns of \( B \) can be interpreted as feature vectors describing the elements of \( S \) and, therefore, \( L \) measures similarity.
Determinantal consensus clustering using dot products between feature vectors. As the dot product is the most natural similarity measure between vectors, it establishes a connection with the well-known kernel-based methods. Kernels are widely used to describe and quantify how any two objects are related. They have been widely used in pattern analysis, like classification and clustering (Howley and Madden 2006). By (2), we can see that the probability assigned by a DPP to a subset $Y$ is related to the volume spanned by its associated feature vectors: sets composed of very diverse elements have higher probabilities, because their feature vectors are more orthogonal, and span larger volumes.

Methods based on “sum-of-squares” have proved to be effective for datasets with ellipsoidal clustered structures (Jain and Dubes 1988). However, if the boundaries that separate clusters are non-quadratic, these methods will fail to generate an effective clustering configuration. One of the several approaches to deal with this problem consists in nonlinearly transforming the data into a high-dimensional feature space, so that clustering analysis can be conducted in this feature space, constructing an optimal separating hyperplane (Vapnik 1995; Girolami 2002; Murua et al. 2008). Kernel-based methods calculate a similarity measure between each pair of elements on the feature space to afterwards use algorithms that only need the value of this measure (Schölkopf et al. 2004).

The choice of an appropriate kernel function $\kappa(x_i, x_j)$, for all $x_i, x_j \in S$, is a critical step in the application of any kernel-based method. Kernel functions are often considered measures of similarity, since a higher kernel value represents a higher correlation in the feature space. Therefore, for the DPP established in (1), we will use $L = (\kappa(x_i, x_j))_{i,j=1}^n$. There is no rule or consensus about the choice of the most suitable kernel function for a particular problem (Howley and Madden 2006). In the absence of expert knowledge, a common choice is the Radial Basis Function (RBF) kernel (or Gaussian kernel):

$$\kappa(x_i, x_j) = \exp\left(-\|x_i - x_j\|^2/(2\sigma^2)\right), \quad (3)$$

where the scale parameter $\sigma$, known as the bandwidth of the kernel, represents the relative spread of the distances $\|x_i - x_j\|$. Here, the distance $\|x_i - x_j\|$ represents the Euclidean distance between $x_i$ and $x_j$, a common choice for the RBF. This kernel can be interpreted as a scaled measure of similarity between $x_i$ and $x_j$. It has been extensively used and studied in machine and statistical learning applications (Girolami 2002). So we have chosen it for building the similarity matrix $L$ of the DPP. The computation of the RBF kernel requires the estimation of the bandwidth parameter $\sigma$. Most of the literature considers $\sigma$ as a parameter that can be estimated by observed data (Murua and Wicker 2014). Inspired by Blatt et al. (1996) and Blatt et al. (1997), we estimate $\sigma^2$ by the average of all pairwise squared Euclidean distances, i.e.,

$$\hat{\sigma}^2 = 2 \sum_{i < j} \|x_i - x_j\|^2/(n(n - 1)). \quad (4)$$

The authors justify the use of the average to estimate $\sigma^2$ based on local structure of the data and identification of high-density regions in the data space. Other methods of
estimating the bandwidth parameter can be found in the literature. Murua et al. (2008) do not consider \(\sigma\) as fixed and propose an adaptive bandwidth selection procedure, where \(\sigma\) depends on the data points. They explore the relationship between Potts model and kernel density estimation, building an algorithm based on Markov Chain Monte Carlo methods to obtain a Bayesian estimate of \(\sigma\). Murua and Wicker (2014) consider \(\sigma\) as fixed and obtain its Bayesian estimate based on the Wang–Landau algorithm (Wang and Landau 2001). Sejdinovic et al. (2013) refer the median of the pairwise Euclidean distances as a common choice. However, Chaudhuri et al. (2017) demonstrate that the use of the average and the median of Euclidean distances to estimate \(\sigma\) produce similar clustering results for the majority of situations. They justify the use of the average distances by its simplicity and fast computation even when the dataset is large. We decided to use the average for the same reasons.

To explore the sensitivity of the clustering configuration to the parameter \(\sigma\), we decided to introduce a tuning parameter \(s > 0\), which will be estimated heuristically by simulation in Sect. 4.2. With the bandwidth estimate given by (4) and the tuning parameter \(s\), the RBF kernel in (3) will be adjusted to

\[
\kappa(x_i, x_j) = \exp \left( -\|x_i - x_j\|^2 / (2s\hat{\sigma}^2) \right).
\] (5)

4 Consensus DPP

In this section we develop a partitioning clustering algorithm that will be run \(R\) times over the set \(S\), in order to obtain a consolidated clustering configuration by consensus clustering. To build a consensus clustering, any partitioning clustering method can be chosen. We propose the use of determinantal point processes as the partition generating algorithm. The algorithm is also based on a Voronoi diagram as described next.

Voronoi diagrams support many clustering techniques (Aurenhammer 1991), such as the \(k\)-means and \(k\)-medoids algorithms, for example. A Voronoi diagram refers to a partition of the space into several cells or regions, based on a subset of elements that are called generator points, or simply generators. Each cell includes only one generator and all the space points that are closer to that generator than to any other generator. For a formal definition of Voronoi diagram, see for example (Okabe et al. 2000).

Let \(\mathcal{P} = \{p_1, \ldots, p_K\} \subseteq S\) be the generator set of the Voronoi diagram. For each \(x \in S\), let \(p_x = \{p_k : \|x - p_k\| \leq \|x - p_j\| \text{ for all } j = 1, \ldots, K\}\) be the set of nearest-generator points to \(x\). In most cases, \(p_x\) is just a singleton. But in rare occasions, \(p_x\) might contain more than one generator point. Due to this latter case, we define \(p_x^*\) as a uniform draw from \(p_x\). In the case of a singleton the draw is always the unique generator point in the singleton. Hence, without loss of generality, we consider the \(p\)-dimensional Voronoi polyhedron associated with \(p_i, i = 1, \ldots, K\) as the region defined by

\[
V(p_i) = \{x \in S : p_x^* = p_i\}.
\]

The set \(\mathcal{V}(\mathcal{P}) = \{V(p_1), \ldots, V(p_K)\}\) is said to be the \(p\)-dimensional Voronoi diagram generated by \(\mathcal{P}\). We call \(p_i\) the generator or generator point of the \(i\)th Voronoi
polyhedron. The Voronoi diagram is a partition of the data \( S \), and hence a clustering of the data. In order to obtain a Voronoi diagram, one needs to select the set of generators. We proposed using a determinantal point process rather than a classical random sampling for this step. We conjecture that sampling from a DPP for Voronoi generators is more efficient than sampling generator points uniformly at random as it is usually done in PAM. Our experiments in Sect. 4.2 corroborate this belief.

Ben Hough et al. (2006) and Kulesza and Taskar (2012) present an efficient scheme to sample from a DPP. The algorithm is based on the following observations. Let \( L = \sum_{i=1}^{n} \lambda_i(L) v_i v_i^T \) be an orthonormal eigendecomposition of \( L \). For any set of indexes \( J \subseteq \{1, 2, \ldots, n\} \), define the subset of eigenvectors \( V_J = \{ v_i : i \in J \} \), and the associated matrix \( K_J = \sum_{i,j \in J} v_i v_j^T \). It can be shown that the matrix \( K_J \) defines a so-called elementary DPP which we denote by \( \text{DPP}(K_J) \). It turn out that the \( \text{DPP}_S(L) \) is a mixture of all elementary DPP given by the index sets \( J \). That is

\[
\text{DPP}_S(L) = \left[ \sum_J \text{DPP}(K_J) \prod_{i \in J} \lambda_i(L) \right] / \det(L + I_n)
\]

The mixture weight of \( \text{DPP}(K_J) \) is given by the product of the eigenvalues \( \lambda_i(L) \) corresponding to the eigenvectors \( v_j \in V_J \), normalized by \( \det(L + I_n) = \prod_{i=1}^{n} [\lambda_i(L) + 1] \). Sampling can be realized by first selecting an elementary DPP, \( \text{DPP}(K_J) \), with probability equal to its mixture component weight, and then, in a second step, sampling \( Y \sim \text{DPP}(K_J) \). In particular, it can be shown that in this case, necessarily \( \text{card}(Y) = \text{rank}(K_J) \), where for any set \( A \), \( \text{card}(A) \) denotes the number of elements in the given set \( A \).

For the purpose of consensus clustering and the construction of the consensus matrix with entries defined in Sect. 2, we will consider \( R \) runs of the sampling algorithm of Ben Hough et al. (2006) and Kulesza and Taskar (2012). The sampling of the \( R \) sets is done from the DPP with associated kernel matrix \( L \) constructed with the RBF kernel in (5). This yields \( R \) generator sets \( \{P_r\}_{r=1}^{R} \). For each generator set, we construct a \( p \)-dimensional Voronoi diagram \( V(P_r) \) based on the similarities given by \( L \). The binary matrix \( C_r \) introduced in Sect. 2 has an entry \( c_{ij}^r = 1 \) if and only if the points \( x_i \) and \( x_j \) fall in the same Voronoi cell, \( r = 1, \ldots, R \). The corresponding consensus matrix \( C \) is finally given by the average of all the matrices \( C_r \) over the \( R \) runs, \( r = 1, \ldots, R \).

The consensus matrix represents the proportion of runs in which two elements \( x_i \) and \( x_j \) of \( S \) belong to the same cluster. The consolidated clustering configuration is obtained by a thresholding procedure. If \( C_{ij} \geq \theta \), with \( 0 \leq \theta \leq 1 \), points \( x_i \) and \( x_j \) are defined as “friends” and then included in the same final cluster (Blatt et al. 1996). Moreover, all mutual friends (including friends of friends, etc.) are assigned to the same cluster. It can be shown that this is equivalent to finding the connected components of the consensus graph introduced in Sect. 2 (Murua et al. 2008). For each threshold \( \theta \), finding the connected components is of order \( O(n \log n) \) (Murua and Wicker 2014).

The choice of the threshold is not an easy task. Although a value of \( \theta = 0.5 \) makes sense most of the time, it might not be the optimal choice. In fact, choosing a fixed and unique threshold does not necessarily give the best clustering results (Murua and
Wicker 2014). Changing the threshold yields different clustering results. Many of those clusterings are worth exploring. Murua and Wicker (2014) consider all threshold values from the set of all different observed consensus indexes $C_{ij}$ (see Sect. 2). If there are $t$ different consensus indexes, we will have a collection of $t$ thresholds $\theta_1, \theta_2, \ldots, \theta_t$. For each threshold $\theta_i, i = 1, \ldots, t$, a consolidated clustering configuration with $K(\theta_i)$ clusters is obtained. If $\theta_i = 0$, we obtain a graph with $K(0) = 1$ cluster, that is, $S$. If $\theta_i = 1$, we obtain a graph with $K(1) = n$ clusters; that is, each element of $S$ is an isolated point and form a singleton cluster of size one. In general, clustering configurations with one cluster or $n$ clusters are of no interest. Therefore, thresholds $\theta_i$ that are too low or too large are not relevant. We adopt a mixed strategy between choosing a predetermined fixed threshold (Blatt et al. 1996) and studying a sequence of interesting thresholds (Murua and Wicker 2014). We consider a sequence of $t$ predetermined thresholds $\theta_1, \theta_2, \ldots, \theta_t$ that are above a certain minimum threshold $\tau$. The value of $\tau = 0.6$ has been determined through simulations. These are reported in Sect. 4.2.

A given sequence of increasing thresholds $\theta_1 < \theta_2 < \cdots < \theta_m$, gives rise to hierarchical cluster partitions. The associated consolidated matrices $C^{(1)}, \ldots, C^{(m)}$ are nested in the sense that when $s < t, C^{(s)}_{ij} \geq C^{(t)}_{ij}$ for all pairs $(i, j) s,t = 1, \ldots, m$. This holds because if $C^{(t)}_{ij} = 1$, the proportion of times that the points $x_i$ and $x_j$ appear together in a partition is at least $\theta_t$. Since $\theta_s \leq \theta_t$, necessarily $C^{(s)}_{ij} = 1$ as well. Similarly, if $C^{(s)}_{ij} = 0$, then necessarily $C^{(t)}_{ij} = 0$. This property of the consolidated matrices implies that the consensus partitions are also nested. The proof of this result is given in the “Appendix”.

We are not interested in a clustering configuration with too many small clusters. We impose a minimal size for each cluster, accepting only clustering configurations with cluster sizes larger than that minimal value. For the establishment of the minimal size, we decided to take a classical approach, inspired by the “square-root choice” for the number of bins of a histogram, $\sqrt{n}$. However, we also consider a more general case that eliminates all clusters with less than $n^a$ elements for a predetermined power $a \in (0, 1)$. The optimal value of the power $a$ depends on various considerations such as the data size, the data dimension, and the number of clusters. We have studied it through the simulations reported in Sect. 4.2. In summary, we examine all the $t$ consolidated clustering configurations obtained with all the different considered thresholds $\theta_1, \theta_2, \ldots, \theta_t$. If one configuration does not satisfy the minimal cluster size criterion, we merge each small cluster with its closest “large” cluster, according to the following procedure: select the component $V$ that has the smallest cluster size $< n^a$; find the pair of indexes $(i^*, j^*) \in \{1, \ldots, n\}$ that satisfies $C_{i^*, j^*} = \max\{C_{ij} : x_i \in V, x_j \notin V\}$; merge the component $V$ to the component that includes $x_{j^*}$; repeat the merging procedure until there are no more connected components with cluster size smaller than $n^a$. This procedure is similar to single-linkage (Ao et al. 2005; Bien and Tibshirani 2011) because this latter merges two clusters based on a minimal distance or maximum similarity. We adopted this single linkage-like procedure instead of complete or average linkage, due to the nature of our consensus clustering which looks for connected components on a graph. As explained above, these are sets of
points that can be joined by a path formed by pairs of points that appear together with a high enough frequency given by a particular threshold. Our merging of a small cluster with a larger one corresponds to descending through the space of thresholds until a path linking the small cluster to the larger one is found.

As mentioned above, the choice of the power $a$, the minimum threshold $\tau$, and the number $R$ of runs of our clustering algorithm, have been determined via simulation (see Sect. 4.2).

4.1 Choosing an optimal clustering

There is a vast literature on measuring the quality and the validity of a clustering configuration. Hennig (2019) gives a nice survey of current available cluster validity measures. We decided to use an internal kernel-based validity measure that only depends on the kernel matrix $L$. Consider the RBF kernel $\kappa(x_i, x_j)$ defined by (3).

The mean scattering induced by the kernel on the data (Vert et al. 2004; Fan et al. 2010), $V_S$, is the average Euclidean distance between the transformed points and the mean of the transformed points. Details on its computation and the computation of the other quantities given in this section can be found in the “Appendix”.

Let $\mathcal{V} = \{V_1, \ldots, V_K\}$ be a clustering configuration with $K$ clusters. Also, let $n_k$ be the size of cluster $V_k$, and let $J_k = \{i : x_i \in V_k\}$, $k = 1, \ldots, K$. As with the mean scattering, the kernel induced scattering $W_{V_k}$ within each cluster $V_k$ is the mean distance in the transformed space between the points in cluster $V_k$ and the corresponding cluster mean (Fan et al. 2010). The average within cluster scattering associated with clustering configuration $\mathcal{V}$ is defined as $W_{\mathcal{V}} = \sum_{k=1}^{K} W_{V_k} / (K V_S)$. To measure the total scattering between clusters, one considers the distance between cluster means in the transformed space $B^2(V_i, V_j) = [B(V_i, V_j)]^2$, and $B_{\mathcal{V}} = (B_{\max} / B_{\min}) \sum_{i=1}^{K} \sum_{j=1}^{K} [B^2(V_i, V_j)]^{-1}$, where $B_{\max} = \max_{(i,j)} B^2(V_i, V_j)$, and $B_{\min} = \min_{(i,j)} B^2(V_i, V_j)$. A simple measure of quality of the clustering configuration is the kernel-based validation index (Fan et al. 2010)

$$KV_{VI_{\mathcal{V}}} = \alpha W_{\mathcal{V}} + B_{\mathcal{V}},$$

where $\alpha$ is a tuning parameter that Fan et al. (2010) set to the value of $B_{\mathcal{V}}$ associated with the largest clustering size $K$ among those clustering configurations being considered. The optimal clustering configuration among the set of all retained clustering configurations is the one that minimizes (6). The steps of the consensus clustering algorithm presented in this section are summarized in Algorithms 1 below and Algorithm 2 of the “Appendix”. Algorithm 2 gives an efficient scheme to sample $Y \sim DPP_{\mathcal{S}}(L)$ and is found in Kulesza and Taskar (2012, Page 145). We adapted it using the notation of our paper. Algorithm 1 permits to obtain a set of $t$ clustering configurations, where each configuration is built from each threshold $\theta_1, \theta_2, \ldots, \theta_t$ applied to the consensus matrix $C$. This matrix is obtained after running Algorithm 2 $R$ times. The optimal clustering configuration is the one among the $t$ clustering configurations that minimizes the kernel-based validation index defined in (6).
Algorithm 1: Consensus DPP

Input: $n \times n$ kernel matrix $L$

$C \leftarrow (0)_{i,j=1}^n$

for $r = 1$ to $R$ do

$P \leftarrow Y_r$ from Algorithm 1

$M_{P} \leftarrow (0)_{i,j=1}^n$

for $i = 1$ to $n$ do

$C_r \leftarrow (0)_{i,j=1}^n$

for $j = 1$ to $n$ do

if $M_{P}[i] = M_{P}[j]$ then

$C_r[i,j] \leftarrow 1$

end

end

end

$C \leftarrow C + C_r$

end

$C \leftarrow C / R$

thresholds $\leftarrow$ values in $C$ greater than 0.6

$V \leftarrow \{1, 2, \ldots, n\}$

foreach $\theta \in$ thresholds do

undirected graph $G = (V, E)$

for $i = 1$ to $n - 1$ do

if $C[i,j] > \theta$ then

$E \leftarrow E \cup$ edge $(i, j)$

end

end

$\mathcal{V}(P) \leftarrow$ connected components of $G$

if any component of $\mathcal{V}(P)$ has size less than $n^a$ then

repeat

select component $\mathcal{V} \in \mathcal{V}(P)$ that has the smallest size

select $C^*_{ij} = \max\{C_{ij} : x_i \in \mathcal{V}, x_j \notin \mathcal{V}\}$ from $C$

merge $\mathcal{V}$ with component that includes $x_j$ corresponding to $C^*_{ij}$

until no component of $\mathcal{V}(P)$ has size less than $n^a$

end

end

$CC \leftarrow$ set of $\text{card(thresholds)}$ clustering configurations

Compute $KV_{I_V}$ measure for each clustering configuration $\mathcal{V}(P) \in CC$

Output: select clustering configuration $\mathcal{V}(P)^*$ with smallest $KV_{I_V}$

4.2 Setting appropriate consensus DPP parameters

In this section, we will conduct simulations to choose the tuning parameters of the proposed clustering method. These are (i) the tuning parameter $a$ for the minimal size of clusters $n^a$; (ii) the number $R$ of sufficient runs of the clustering algorithm; (iii) the inferior limit $\tau$ for the threshold range to obtain the consolidated clustering.
configurations; and (iv) the tuning parameter $s$ in (5) for the sensitivity of the bandwidth parameter $\hat{\sigma}^2$.

**Data generation.** The simulated data were generated with the algorithm of Melnykov et al. (2012). This generates datasets from $p$-variate Gaussian mixtures with $K$ components, $\sum_{k=1}^{K} \pi_k \phi_p(\cdot; \mu_k, V_k)$, where $\phi_p$ denotes the $p$-variate normal density. The mean vectors of the components, $\{\mu_1, \ldots, \mu_K\}$, are obtained as $k$ independent realizations from a uniform $p$-variate unit hypercube. The covariance matrix of each component, $V_k$, is obtained as a realization from the $p$-variate standard Wishart distribution with $p + 1$ degrees of freedom. The mixing proportions $\pi_k$ are generated from a Dirichlet distribution on the standard $K - 1$ simplex, so that $\sum_{k=1}^{K} \pi_k = 1$. The number of elements generated from each component is obtained from the multinomial distribution based on the mixing proportions.

The algorithm allows to control the *pairwise overlap* between two components, which measures the interaction between components, and controls the clustering complexity of datasets simulated from the mixtures. For the purposes of our study, we focus on low pairwise overlap cases, as the notion of clustering itself becomes less meaningful as the overlap degree between clusters becomes large. An average overlap of 0.4 is considered extreme while an average overlap of 0.001 is considered very low (Melnykov et al. 2012). We fix a maximum pairwise overlap of 0.01 between any two components. The values of the pairwise overlap were then generated uniformly at random taking into account this constraint. Moreover, the data were obtained from mixtures with ellipsoidal covariance matrices, and unequal number of elements per component.

We studied the effect of three variables on the clustering results. These are the number of observations per dataset $n \in \{150 \text{ (low)}, 500 \text{ (medium)}, 1500 \text{ (large)}\}$; the number of variables per dataset $p$: low ($2 \leq p \leq 7$), medium ($8 \leq p \leq 12$) and large ($13 \leq p \leq 20$); and the number of components or clusters per dataset $K$: low ($2 \leq K \leq 5$), medium ($6 \leq K \leq 10$) and large ($11 \leq K \leq 20$). The values of $p$ and $K$ were chosen randomly in each case once the level (low, medium or large) was chosen. The three categories generate a $3^3$ factorial design with 27 experimental conditions. To simulate data for this factorial design, we ensured that no cluster with a small number of elements was present in the simulated datasets. This was done because, according to the merging procedure described in Sect. 4, small clusters would be inevitably merged with a larger cluster. This consideration results in simulated data with more or less balanced clusters. However, as it is impossible to simulate a dataset with a low number of observations ($n = 150$) and a large number of clusters ($11 \leq K \leq 20$) that contains no cluster with a small number of elements, this case has been excluded from our analysis. Therefore, we only consider 24 experimental conditions of the $3^3$ factorial design, generating 10 datasets per condition, obtaining a total of 240 datasets. We will refer to the 24 experimental conditions as the 24 experimental scenarios, or scenarios, for short.

**Measuring the quality of the clustering.** In order to measure the quality of the clustering results we use the Adjusted Rand Index (ARI), which is a common measure of goodness-of-fit in the clustering literature (Yeung et al. 2001; Murua and Wicker 2014). The ARI was first introduced by Rand (1971) and later adjusted for randomness...
by Hubert and Arabie (1985). It is a measure of agreement between two clustering configurations. The original Rand Index counts the proportion of elements that are either in the same clusters in both clustering configurations or in different clusters in both configurations. The adjusted version of the Rand Index corrected the calculus of the proportion, so that its expected value is zero when the clustering configurations are random. The larger the ARI, the more similar the two configurations are, with the maximum ARI score of 1.0 indicating a perfect match.

We will also use the relative difference in the estimated number of clusters and the true number of clusters \(RN = (\sqrt{\hat{G}} - \sqrt{G})/\sqrt{G}\), where \(\hat{G}\) is the estimated number of clusters and \(G\) is the true number of clusters (Muñoz and Murua 2018). Here, small absolute values of RN are preferred and, therefore, the absolute value of RN will be used as another measure of clustering quality.

Results. For each of the ten replicas associated with one of the 24 scenarios, we run consensus DPP with \(R = 1000\), and \(a \in \{1/4, 1/3, 1/2, 2/3\}\). To obtain the consolidated clustering configurations with the thresholding procedure, we set the threshold to a unique fixed value in \{0.4, 0.5, 0.6, 0.7, 0.8, 0.9\}. The quality of each clustering configuration associated with each threshold was assessed with the ARI criterion, after the first 10, 50, 100, 200, 300, 500, 700, 900 and 1000 runs. As each case is evaluated on the ten replica datasets of a particular scenario, the ARIs were averaged over the ten runs. For the choice of the optimal clustering, we used the kernel-based validation index, defined in (6).

From the results (not shown here) we observe a relative stability of the ARI mean values for most of the cases. Also, setting \(a = 1/2\) for the minimal cluster size criterion of the merging procedure seems a reasonable choice, which corresponds to the classic choice of \(\sqrt{n}\). With respect to the number of runs required to achieve a good clustering fit, the simulations show that a number of runs between 50 and 200 is adequate. As the best results are obtained most of the times with 200 runs, we adopt \(R = 200\). The choice of the inferior limit \(\tau\) for the range of thresholds depends slightly on the experimental scenario. The optimal values of \(\tau\) are larger for datasets with a large number of clusters. In this scenario, values of 0.7–0.9 are preferred. For the other cases, the optimal value hovers around 0.5. In order to recommend a unique value, we took the average among the optimal values of \(\tau\) for each scenario. This yielded \(\tau = 0.6\).

To evaluate the sensitivity of the bandwidth parameter \(\hat{\sigma}^2\) through the tuning parameter \(s\) in (5), we evaluate the ARI criterion as above for every value of \(s \in \{0.5, 0.75, 1, 1.25, 1.5, 2\}\). Following the above observations on the optimal values for \(a\), \(R\) and \(\tau\), for this experiment, we set \(a = 0.5\), \(R = 200\), and \(\tau = 0.6\). The simulation results clearly show that the value of \(s\) has almost no effect on the clustering configurations, nor the corresponding ARI values. Therefore, we decided to fix \(s = 1\) in (5).

We would like to stress that the parameter values chosen as optimal in our study would not necessarily be optimal for all situations. However, they could be taken as good values to start exploring the data at hand.
5 Comparison performance between consensus DPP and PAM

As the choices for $a$, $R$, $\tau$ and $s$ have been fixed, we can now assess the performance of our clustering algorithm. For the evaluation, we use the procedure to simulate data described in the previous section. As a reference, and for comparison purposes, our consensus clustering methodology will not only be applied to clustering configurations generated with DPP, but also to clustering configurations generated by the well-known Partitioning Around Medoids (PAM) method. The PAM method is a classical partitioning technique for clustering introduced by Kaufmann and Rousseeuw (1987). These authors proposed a sequential process to optimally select the initial set of points. In order to simplify the selection process of PAM, several authors have suggested the Voronoi iteration method whose centers are chosen by simple random sampling (Park and Jun 2009; Ibrahim and Harbi 2013; Budiaji and Leisch 2019; Schubert and Rousseeuw 2019). For comparison purposes, we use this latter approach. We select data points centers by simple random sampling, to then apply our consensus clustering methodology to build the final clustering configuration. This procedure, previously introduced in Chen et al. (2009), will be denoted by Consensus Clustering-PAM (CC-PAM). As DPP selects data points centers based on diversity, our goal here is to study how the quality of the clustering configurations depends on diversity at sampling centroids.

To sample points with the CC-PAM procedure, we first sample a number $k$ of Voronoi cells uniformly at random from a finite set of integers $\{1, \ldots, K_{\text{max}}\}$. Then, we sampled uniformly at random $k$ points from the dataset. Although, the sampling is uniform over the subset sizes, and over subsets of the same size, this sampling technique is not really uniform; it favors very large and very small subsets over moderately sized subsets. In fact, the probability of choosing a subset of $k$ points $\{x_{i1}, \ldots, x_{ik}\}$ following this sampling technique is $\left(K_{\text{max}} \binom{n}{k}\right)^{-1}$. Despite this fact, we will refer to this sampling as uniform random sampling.

In what follows, we first compare the performance of DPP and CC-PAM on data with elliptical (Gaussian) clusters. Then, we study the sensitivity of DPP with respect to the kernel chosen to measure similarities between data points. Finally, we compare the performance of DPP and CC-PAM on data with non-Gaussian clusters.

5.1 Gaussian clusters

First, we compare the ARI trajectories of consensus DPP with those of CC-PAM as a function on the number of runs $R$. Figure 1 displays these trajectories considering all experimental scenarios described in the previous section. The trajectories globally reflect the performance of each experimental condition involving the three levels of variables (low, medium and large) and the three levels of clusters (low, medium and large). Observe that due to the diversity in the sampling of data points center, DPP has a jump-start like behavior, requiring very few runs to achieve good clustering configurations. CC-PAM, on the contrary, improves slowly its clustering configurations, and sometimes, even after 1000 runs, it never catches up with consensus DPP.
We can see clear benefits of using DPP as a sampling method for the initial points needed to construct the Voronoi diagrams that define the clustering configurations. However, consensus DPP and CC-PAM yield similar results when the variable dimensions are low. The dimension $p$ seems to play a crucial role in the potential of DPPs to sample with more diversity: there are more possibilities of distinguishing two vectors $x_i, x_j \in \mathbb{R}^p$ in the transformed space $\mathcal{H}$ when the dimension $p$ is already large, since the two vectors are more likely to be projected in very different places. In this case, DPP will sample these two points together more often than uniform random sampling, which will sample any pair of points with the same probability. When $p$ is small, the two vectors have less potential of being very different, and using DPP or CC-PAM should give similar results. DPP offers clear benefits when compared to CC-PAM when $p$ is moderate and large.

To complement the differences between DPP and CC-PAM, we show in Fig. 2 typical histograms of the logarithm of the probability mass function of the DPP, given by (1), for 1000 sampled random subsets, using either the DPP sampling algorithm of Ben Hough et al. (2006) and Kulesza and Taskar (2012), or the uniform random sampling of CC-PAM, for three simulated datasets selected among the 24 experimental scenarios.

The histograms clearly show that DPP selects random subsets with higher and less dispersed probability mass values (likelihood) than uniform random sampling. This explains the observed lower dispersion of the ARI values when sampling is performed with DPP. The higher likelihood of the random subsets sampled by DPP confirms the higher diversity of those subsets. Instead, subsets sampled as in CC-PAM can be highly or poorly diverse: in fact, the associated histograms show very high dispersion in terms of diversity. DPP tends to select points that maintain a high level of diversity at each
Next, we fix $R = 200$ as suggested by the study of the previous section, and compare the performance of consensus DPP and CC-PAM. We have already noted the superiority of consensus DPP when looking at ARI as a measure of clustering quality. But, this time we apply our mixed strategy presented in Sect. 4. That is, we consider a sequence of thresholds $\theta_1, \theta_2, \ldots, \theta_t$ that are above the inferior limit $\tau$. We recall that adopting the strategy of considering a range of thresholds results in a collection of consolidated clustering configurations for each datasets in each scenario. For the choice of the optimal clustering configuration, we use the kernel-based validation index $KVI_V$ in (6). To measure the goodness-of-fit of the optimal clustering configuration we use the ARI and RN measures described earlier. Table 1 displays the ARI means and standard deviations over all 24 scenarios, while Table 2 displays the RN means and standard deviations over all 24 scenarios.

As noted earlier, there is a clear advantage of using DPP over uniform random sampling for the initial points needed to construct the Voronoi diagrams. This is particularly true when the number of variables is moderate to high. When the number of variables is low, the results yielded by DPP and CC-PAM are similar. Another interesting advantage that we observe is that sampling with DPP contributes to reducing the dispersion of the ARI scores, and then produces more stable optimal clustering configurations. Turning now to the RN means of Table 2, the same conclusion applies: sampling with DPP yields better results. The optimal clustering configurations yielded by consensus DPP are associated with estimated number of clusters closer to the true number of clusters than those yielded by CC-PAM. Observe as well, that DPP contributes to the reduction of the variability of the RN values for almost all the cases, as it was already the case with ARI.

### 5.2 Non-Gaussian clusters

In this section, we evaluate the performance of consensus DPP, compared to CC-PAM and $k$-means on data composed of non-Gaussian clusters. The $k$-means algorithm was proposed by Stuart Lloyd in 1957, and later published in Lloyd (1982). It starts with
### Table 1  
ARI means and standard deviations (within parentheses) over all 24 scenarios with consensus DPP and CC-PAM

| Sample size | Method | Number of clusters | Number of variables | Low      | Medium | Large     |
|-------------|--------|--------------------|--------------------|----------|--------|-----------|
|             |        |                    |                    |          |        |           |
|             | DPP    | Low                | 0.95 (0.07)        | 0.92 (0.16) | 0.90 (0.15) |
|             | CC-PAM |                    | 0.90 (0.14)        | 0.93 (0.13) | 0.74 (0.22) |
|             | DPP    | Medium             | 0.92 (0.09)        | 0.94 (0.07) | 0.85 (0.10) |
|             | CC-PAM |                    | 0.98 (0.02)        | 0.83 (0.14) | 0.65 (0.14) |
| n = 150     | DPP    | Low                | 0.95 (0.11)        | 0.95 (0.10) | 0.92 (0.09) |
|             | CC-PAM |                    | 0.77 (0.22)        | 0.86 (0.19) | 0.83 (0.18) |
|             | DPP    | Medium             | 0.95 (0.08)        | 0.98 (0.03) | 0.91 (0.11) |
|             | CC-PAM |                    | 0.96 (0.08)        | 0.90 (0.11) | 0.71 (0.34) |
|             | DPP    | Large              | 0.96 (0.05)        | 0.96 (0.03) | 0.98 (0.02) |
|             | CC-PAM |                    | 0.99 (0.02)        | 0.87 (0.10) | 0.70 (0.15) |
| n = 500     | DPP    | Low                | 0.95 (0.11)        | 0.95 (0.10) | 0.92 (0.09) |
|             | CC-PAM |                    | 0.77 (0.22)        | 0.86 (0.19) | 0.83 (0.18) |
|             | DPP    | Medium             | 0.95 (0.08)        | 0.98 (0.03) | 0.91 (0.11) |
|             | CC-PAM |                    | 0.96 (0.08)        | 0.90 (0.11) | 0.71 (0.34) |
|             | DPP    | Large              | 0.96 (0.05)        | 0.96 (0.03) | 0.98 (0.02) |
|             | CC-PAM |                    | 0.99 (0.02)        | 0.87 (0.10) | 0.70 (0.15) |
| n = 1500    | DPP    | Low                | 0.91 (0.08)        | 0.89 (0.11) | 0.90 (0.10) |
|             | CC-PAM |                    | 0.66 (0.26)        | 0.74 (0.18) | 0.76 (0.26) |
|             | DPP    | Medium             | 0.96 (0.04)        | 0.99 (0.01) | 0.88 (0.15) |
|             | CC-PAM |                    | 0.98 (0.02)        | 0.99 (0.01) | 0.96 (0.06) |
|             | DPP    | Large              | 0.97 (0.04)        | 0.96 (0.05) | 0.96 (0.02) |
|             | CC-PAM |                    | 0.93 (0.12)        | 0.90 (0.17) | 0.69 (0.35) |

### Table 2  
RN means and standard deviations (within parentheses) over all 24 scenarios with consensus DPP and CC-PAM

| Sample size | Method | Number of clusters | Number of variables | Low      | Medium | Large     |
|-------------|--------|--------------------|--------------------|----------|--------|-----------|
|             |        |                    |                    |          |        |           |
|             | DPP    | Low                | 0.03 (0.06)        | 0.04 (0.08) | 0.04 (0.06) |
|             | CC-PAM |                    | 0.04 (0.08)        | 0.03 (0.08) | 0.09 (0.12) |
|             | DPP    | Medium             | 0.03 (0.05)        | 0.02 (0.03) | 0.04 (0.04) |
|             | CC-PAM |                    | 0.00 (0.00)        | 0.08 (0.08) | 0.17 (0.07) |
| n = 150     | DPP    | Low                | 0.05 (0.13)        | 0.04 (0.13) | 0.04 (0.08) |
|             | CC-PAM |                    | 0.23 (0.21)        | 0.08 (0.11) | 0.13 (0.17) |
|             | DPP    | Medium             | 0.03 (0.06)        | 0.02 (0.05) | 0.04 (0.07) |
|             | CC-PAM |                    | 0.02 (0.06)        | 0.05 (0.06) | 0.16 (0.20) |
|             | DPP    | Large              | 0.02 (0.02)        | 0.02 (0.03) | 0.01 (0.01) |
|             | CC-PAM |                    | 0.01 (0.01)        | 0.06 (0.06) | 0.14 (0.06) |
| n = 500     | DPP    | Low                | 0.11 (0.13)        | 0.16 (0.21) | 0.18 (0.22) |
|             | CC-PAM |                    | 0.50 (0.40)        | 0.37 (0.28) | 0.44 (0.50) |
|             | DPP    | Medium             | 0.02 (0.04)        | 0.00 (0.00) | 0.16 (0.21) |
|             | CC-PAM |                    | 0.01 (0.03)        | 0.00 (0.00) | 0.03 (0.08) |
|             | DPP    | Large              | 0.02 (0.02)        | 0.02 (0.03) | 0.01 (0.03) |
|             | CC-PAM |                    | 0.03 (0.06)        | 0.04 (0.07) | 0.14 (0.16) |
an initial set of $k$ means, representing $k$ clusters. It assigns each observation to the corresponding Voronoi cell or cluster given by the corresponding closer mean among the $k$ means. Once all observations are assigned, the mean vectors of all Voronoi cells are updated, and the process is repeated until there is no change in the means. However, the popular methods for choosing the initial set of $k$ means, such as Forgy (Forgy 1965), Random Partition (Pena et al. 1999) and Maximin methods (Gonzalez 1985; Katsavounidis et al. 1994), result often in cluster configurations with a low clustering quality (Celebi et al. 2013). For that reason we decided to work with the $k$-means++ algorithm (Arthur and Vassilvitskii 2007), a popular choice mentioned by several authors (Capó et al. 2017; Fränti and Sieranoja 2019) that avoids the poor quality results of the traditional methods for choosing the initial means. It is based on a simple probabilistic technique. For the consensus clustering with $k$-means, we proceed as follows: we first sample a number $k$ of Voronoi cells uniformly at random from a finite set of integers $\{1, \ldots, K_{\text{max}}\}$. Then we run $k$-means++ to obtain an initial set of $k$ means. This step consists of (i) selecting the first center at random from the dataset $S$; and (ii) repeating the following two steps until a subset of $k$ centers has been sampled: (a) for each $x \in S$, we compute $dc^2(x)$ the square of the Euclidean distance between $x$ and the closest center among those already sampled; (b) a new center is sampled with probability $dc^2(x)/\sum_{x' \in S} dc^2(x')$. Once the $k$ centers have been chosen, we proceed as in the standard $k$-means algorithm described above. We repeat the selection of $k$ initial means $R = 200$ times, just as we do with consensus DPP and CC-PAM, to afterwards apply our consensus clustering methodology to the clustering configurations. We will denote the whole procedure by Consensus Clustering-$k$-means++ (CC-$k$-means++).

We used the simulated datasets described in Sect. 4.2, which consist in ten replicas of the 24 scenarios described in the same section. Recall that the experimental conditions considered were $n \in \{150 \text{ (low)}, 500 \text{ (medium)}, 1500 \text{ (large)}\}$, $p \in \{5 \text{ (low)}, 10 \text{ (medium)}, 14 \text{ (large)}\}$, and $K \in \{3 \text{ (low)}, 10 \text{ (medium)}, 18 \text{ (large)}\}$. In order to obtain non-ellipsoidal (non-Gaussian) clusters, we selected one of the ten replicas and transformed the data on each of the 24 scenarios with the inverse Box–Cox transformation (Box and Cox 1964). For the power parameter $\lambda$ of the Box–Cox transformation, we selected $\lambda = 1/3$, $\lambda = 1/2$, and $\lambda = 2$. We ran consensus DPP, CC-PAM and CC-$k$-means++ on the transformed data as described above, running five repetitions of each algorithm. We computed the ARI of the five final clustering configurations for each algorithm. Figure 3 shows a barplot of the average ARIs for the case of large datasets ($n = 1500$). Their corresponding standard errors are drawn as segments measuring the average ARI plus and minus one standard error. The results associated to medium ($n = 500$) and small ($n = 150$) datasets are similar and not shown here. There is not much difference in performance when $p$ is small, except when $K$ is also small. In this latter case, DPP appears to perform better than the other two methods. The differences in performance can be seen when $p$ increases, specially when $K$ is large for all three transformations. We can see that CC-$k$-means++ and CC-PAM are more sensible to the complexity of the data, that is, when $p$ and $K$ increase, than DPP. The performance of DPP is pretty stable across all scenarios studied. This DPP behavior is very similar to the one observed in the simulations involving Gaussian clusters (see Table 1).
We also carried out a sensitivity analysis on the consensus DPP methodology. The goal was to study the effect on the quality of the final clustering configuration under different DPP kernels, and under different estimates of the bandwidth. We also investigated the behavior of the kernels on data whose clusters are non-Gaussian, and compare the results with those produced by CC-PAM and CC-k-means++. We considered three kernels: the linear kernel, $\kappa(x_i, x_j) = x_i^T x_j$, and two variants of the $q$-stable kernel (Bilodeau and Nangue 2017), also known as the $q$-Gaussian kernel (Verleysen and François 2005), which is given by $\kappa(x_i, x_j) = \exp\left(- \frac{\|x_i - x_j\|^q}{\sigma^q}\right)$, $q \in (0, 2]$, where $\sigma > 0$ is the kernel bandwidth. The variants considered are the Laplace kernel ($q = 1$), and the Gaussian kernel ($q = 2$). Further details on the simulations and results are given in the “Appendix” (see Fig. 5 of the “Appendix”).

We observed that the linear kernel is not a good option, as it yields the lowest ARI values. The Gaussian kernel is the DPP kernel that shows the highest ARI values. The Laplace kernel yields intermediate results: while some results are comparable to the ARI obtained with the Gaussian kernel, others are clearly inferior (specially when $K$ is small). We also observed that DPP with Gaussian kernel is very stable across data with very different cluster shapes. Overall, DPP with the Gaussian kernel performs better than CC-PAM and CC-k-means++. The choice of kernel does have an effect in the DPP results. If chosen correctly, consensus clustering with DPP has the potential to outperform other methods across different data scenarios. An advantage of DPP
Table 3  Selected datasets from the UCI and openML repositories

| Dataset     | n    | K | p  | Dataset     | n    | K | p  |
|-------------|------|---|----|-------------|------|---|----|
| Iris        | 150  | 3 | 4  | Forest      | 198  | 4 | 27 |
| OliveOil    | 572  | 9 | 8  | Breast      | 569  | 2 | 30 |
| Ecoli       | 327  | 5 | 7  | Synthetic   | 600  | 6 | 60 |
| Bank        | 1372 | 2 | 4  | Lung cancer | 181  | 2 | 12,533 |
| Colposcopy  | 287  | 3 | 62 | Yeast Cycle | 384  | 5 | 17 |

is that depending on the data at hand, the kernel can be chosen accordingly. This is demonstrated in the next section with an application to document clustering.

6 Application to real data

In this section we proceed to evaluate the performance of consensus DPP versus CC-PAM on real datasets. The datasets were obtained from the UCI Machine Learning Repository (Dua and Graff 2017) and OpenML website (Vanschoren et al. 2013), two well known databases in the Machine Learning community for clustering and classification problems. Table 3 shows the selected real datasets and some of their features: $n$ = number of observations, $K$ = number of clusters, $p$ = number of variables (i.e., data dimension).

Following the recommendations in Bicego and Baldo (2016) and Xuan et al. (2013), and due to its strongly unbalanced nature, the Ecoli dataset was transformed using the Box–Cox transformation procedure. Moreover, the original dataset contains $n = 336$ observations with $K = 8$ clusters, but two clusters have only 2 observations, and a third cluster has only 5 observations. These clusters were removed from the data. The Breast and Lung Cancer datasets were also transformed using the Box–Cox transformation. We note that transforming the data is a common procedure for DNA microarray data (Thygesen and Zwinderman 2004). The Bank dataset has only two clusters, even though it contains $n = 1372$ observations. So using $\sqrt{n} \approx 37$ as a minimal cluster size in the cluster merging stage of the consensus procedure is not optimal. We note that our experiments to select the appropriate parameters for consensus DPP hinted at larger values of the power $a$ when the number of clusters is small. Hence, for these data, we used $n^{2/3} \approx 124$ as the minimal cluster size.

For each dataset, we performed $R = 200$ runs of consensus DPP. The procedure was repeated ten times. For the choice of the optimal clustering configuration, we use the kernel-based validation index $KVI_V$ defined in (6). To measure the goodness-of-fit of the optimal clustering configuration we use the ARI and RN measures. We compare the consensus DPP results to those obtained with the application of our consensus clustering methodology to random initial centers generated as in PAM, $k$-means, and kernel $k$-means algorithms. Our goal is to show the advantages of the DPP diversity at sampling centroids on the quality of clustering configurations.
The PAM algorithm was already mentioned in the study with the simulated datasets of Sect. 5. We denoted by CC-PAM our consensus clustering methodology applied to random initial sets of points generated as in PAM algorithm. Our implementation of the consensus k-means algorithm was described in Sect. 5.2. We denoted it by CC-k-means++. The same consensus clustering procedure was applied with kernel k-means++ with a Gaussian kernel (Schölkopf et al. 1998; Arthur and Vassilvitskii 2007). We used the implementation of kernel k-means++ provided by the command kkmeans of the kernlab package (Karatzoglou et al. 2004) of R. As we did with consensus DPP, the optimal cluster configurations from CC-PAM, CC-k-means++, and CC-kernel-k-means++ were chosen using the kernel-based validation index KVIc criterion defined in (6). The whole procedure was repeated ten times. Table 4 displays the ARI and RN means and standard deviations obtained by applying consensus DPP, CC-PAM and CC-k-means++ to the datasets of Table 3.

We observe that, in general, consensus DPP yields higher ARI values than the three other methods, and contributes to reducing the variability of this measure, as well. That is, consensus DPP produces stable and better clustering configurations. The RN results are more balanced in the sense that there is no major difference between the four

| Dataset         | Measure | DPP     | CC-PAM  | CC-k-means++ | CC-kernel-k-means++ |
|-----------------|---------|---------|---------|--------------|---------------------|
| Iris            | ARI     | 0.91 (0.03) | 0.83 (0.09) | 0.66 (0.05) | 0.73 (0.12)        |
|                 | RN      | 0.03 (0.07) | 0.06 (0.08) | 0.02 (0.05) | 0.02 (0.05)        |
| OliveOil        | ARI     | 0.72 (0.06) | 0.60 (0.11) | 0.68 (0.08) | 0.71 (0.06)        |
|                 | RN      | 0.12 (0.05) | 0.10 (0.07) | 0.11 (0.03) | 0.11 (0.06)        |
| Ecoli           | ARI     | 0.76 (0.02) | 0.66 (0.09) | 0.71 (0.07) | 0.75 (0.02)        |
|                 | RN      | 0.05 (0.06) | 0.14 (0.08) | 0.04 (0.05) | 0.06 (0.05)        |
| Bank            | ARI     | 0.66 (0.09) | 0.53 (0.19) | 0.50 (0.10) | 0.23 (0.03)        |
|                 | RN      | 0.13 (0.12) | 0.25 (0.19) | 0.24 (0.15) | 1.34 (0.11)        |
| Colposcopy      | ARI     | 0.44 (0.09) | 0.42 (0.14) | 0.35 (0.11) | 0.43 (0.10)        |
|                 | RN      | 0.21 (0.09) | 0.15 (0.10) | 0.16 (0.14) | 0.19 (0.14)        |
| Forest          | ARI     | 0.86 (0.05) | 0.70 (0.01) | 0.72 (0.22) | 0.72 (0.05)        |
|                 | RN      | 0.01 (0.04) | 0.13 (0.00) | 0.08 (0.12) | 0.13 (0.05)        |
| Breast          | ARI     | 0.61 (0.05) | 0.50 (0.13) | 0.61 (0.13) | 0.23 (0.06)        |
|                 | RN      | 0.09 (0.12) | 0.13 (0.12) | 0.07 (0.11) | 1.17 (0.16)        |
| Synthetic       | ARI     | 0.69 (0.02) | 0.66 (0.04) | 0.64 (0.01) | 0.62 (0.03)        |
|                 | RN      | 0.09 (0.10) | 0.16 (0.08) | 0.13 (0.07) | 0.19 (0.12)        |
| Lung cancer     | ARI     | 0.89 (0.14) | 0.84 (0.34) | 0.65 (0.42) | 0.03 (0.10)        |
|                 | RN      | 0.09 (0.12) | 0.00 (0.00) | 0.02 (0.07) | 0.11 (0.18)        |
| Yeast Cycle     | ARI     | 0.47 (0.004) | 0.47 (0.03) | 0.44 (0.05) | 0.48 (0.01)        |
|                 | RN      | 0.08 (0.06) | 0.10 (0.04) | 0.03 (0.05) | 0.06 (0.05)        |
methods. This means that all four methods hinted at reasonable number of clusters, but not all got good clustering configurations.

As we also did for the simulated data in Sect. 5, we present in Fig. 4 two typical histograms of the logarithm of the probability mass function of the DPP, given by (1). The datasets in the figure are Iris and Synthetic (see Table 3).

As with simulated datasets, we observe that while subsets sampled at random as in CC-PAM result in histograms with a very high dispersion in terms of diversity, DPP tends to select points that maintain a high level of diversity in each sample. DPP seems to be more consistent and stable than uniform random sampling in ensuring the heterogeneity of the elements forming the subsets.

6.1 An application to text mining

In this section, we consider the task of document clustering. The kernel for this application is not a Gaussian kernel, but a dedicated kernel function for text mining. The kernel matrix $L$ was constructed using a string kernel (Watkins 1999; Herbrich 2001) that measures the similarity between two strings $x$ and $\tilde{x}$ by

$$k(x, \tilde{x}) = \sum_{s \in S} \text{num}_s(x) \text{num}_s(\tilde{x})\lambda_s,$$

where $S$ represents the set of all non empty strings, $\text{num}_s(x)$ represents the number of times the substring $s$ occurs in $x$, and $\lambda_s$ is a decay factor whose value may depend on the properties of $s$, such as its length. In this application we used the so-called $\ell$-spectrum string kernel. This string kernel considers all substrings $s$ of exact length $\ell$ in the collection of string sequences $S$ (that is, the documents being analyzed). In this case $\lambda_s$ is a constant.

The data consist of a sample of 40 text documents selected from the Reuters news data set (Lewis 1997). The sample is available in R in the kernlab package. To evaluate the string kernel, we used the stringdot function, also in the kernlab
Table 5  ARI means and standard deviations (within parentheses) associated with consensus DPP, CC-PAM and CC-kernel-k-means, with the length of the substrings ranging from 2 to 7

| length | DPP      | CC-PAM    | CC-kernel-k-means |
|--------|----------|-----------|-------------------|
| 2      | 0.66 (0.05) | 0.54 (0.10) | 0.20 (0.12)       |
| 3      | 0.77 (0.05) | 0.76 (0.13) | 0.24 (0.08)       |
| 4      | 0.34 (0.08) | 0.50 (0.14) | 0.14 (0.05)       |
| 5      | 0.50 (0.05) | 0.35 (0.13) | 0.02 (0.06)       |
| 6      | 0.42 (0.05) | 0.16 (0.11) | 0.03 (0.03)       |
| 7      | 0.53 (0.08) | 0.44 (0.22) | 0.06 (0.09)       |

For each length, the highest ARI among the three methods is highlighted in bold.

package. The obtained kernel values were normalized to remove any bias introduced by document length. We decided to evaluate the sensitivity of the results according to the length $\ell$ of the substrings $s \in S$. We replicated our consensus DPP algorithm five times, with $R = 200$ runs at each time, and computed the ARI for each replica. For comparison purposes, we applied the same procedure with CC-PAM and CC-kernel-k-means. The results are summarized by the ARI means and the corresponding standard deviations, which can be found in Table 5.

7 Conclusions and discussion

We explored the potential of determinantal point processes as a sampling method for initializing each run of a consensus clustering algorithm. As a probabilistic model of repulsion, it favors diversity within subsets of points. This is in contrast to uniform random sampling, which gives to every point an equal probability of being selected. Our simulations showed that, when compared to uniform random sampling, the use of DPPs results in final clustering configurations with higher and less dispersed quality scores. Applications to real datasets confirm these conclusions.

By using DPPs to generate center point sets for clustering, the consensus clustering does not require a large number of sampled partitions to ensure a high goodness-of-fit score in the final clustering configurations. In fact, a moderate number of ensemble partitions of about 100 or 200 is sufficient. In contrast, uniform random sampling generally requires a larger number of sampled partitions to reach ARI mean values comparable to determinantal consensus clustering.

There is limited literature dealing with experimental designs to choose the initial center points. Maitra (2009) suggest a five-stage procedure that specifies initial values by estimating a large number of potential local modes of the distribution associated with the dataset, so as to choose a representative set of the most separated ones. The first step consists in a dimension reduction obtained with a singular value decomposition, so as to search for local models in one-dimensional spaces. The rationale of this procedure is the same as the one driving DPP, which select a set of initial values while maximizing diversity. However, the multistage nature of Maitra (2009) may be seen as a drawback of the approach. Instead, sampling with DPP is a quicker procedure that does not imply several steps. The information on point diversity is captured in both the kernel and the
repulsive nature of DPP. Also, it is not clear that a set composed of a Cartesian product of univariate local-modes would yield local-modes of the multivariate distribution of the data. Another approach that goes in the same spirit as ours, is the approach of Katsavounidis et al. (1994). A set of points is chosen sequentially as the points that are the further apart from each other, starting with the point with the highest Euclidean norm. While the approach is simple and intuitive, it suffers from major drawbacks, when compared to DPP. First, contrary to DPP, the chosen set of initial points is not random, but deterministic. To introduce randomness within the procedure described by the authors, a resampling procedure such as bootstrap (Efron and Tibshirani 1994) could be used. Second, the distances between vectors are computed in the original space of the points. Therefore, this approach does not benefit from the advantages of the kernel trick. If the separation or relationship between points in the original space are non-linear, computing linear distances between points can give poor results. The kernel trick, typical of kernel methods such as DPP, permits to compute linear distances in a space of a superior dimension, where non-linear relationships of the original space are converted in linear relationships in the transformed space.

Selecting an appropriate threshold during the merging procedure of the determinantal consensus algorithm is essential. Our simulations show that a good strategy consists in choosing a small subset of diverse thresholds among all the possible threshold values given by the observed consensus indexes. The main advantage of this strategy is to speed up the computations, while preserving the properties associated with keeping all threshold values from the set of all different observed consensus indexes (Murua and Wicker 2014). Retaining thresholds above 0.6 was adopted as a general choice. For the choice of the final clustering configuration among several candidates, the kernel-based validation index of Fan et al. (2010) has proven to be a good option.

Even if DPP offers clear benefits when compared to CC-PAM when $p$ is moderate to large, some adjustment in the use of the kernels may be necessary for very large $p$. According to Verleysen and François (2005), the “curse of dimensionality” may affect Gaussian kernels which they show, are not able to discriminate between close and far points in high-dimensional spaces, specially when $p = 100$. Our simulations were conducted with $2 \leq p \leq 20$, so we have not detected any problems with the Gaussian kernel. However, it is clear that an extension of our methodology to high-dimensional data would need to consider the curse of dimensionality. Following the work of Verleysen and François (2005), we could use a $q$-stable kernel, $q \in \mathbb{R}^+$, instead of the usual Gaussian kernel (which is a 2-stable kernel), where besides the kernel-bandwidth, $q$ becomes another parameter of the kernel.

A variety of interesting questions remain for future research: (i) To extend the determinantal consensus clustering to datasets with both continuous and categorical variables, inducing the choice of a proper measure of distance, necessary for the construction of the kernel matrix. In general, for continuous random variables, the Euclidean or Mahalanobis-like distances perform well. For categorical data, Lin’s pairwise similarity measure (Lin 1998) is an attractive alternative to the usual Hamming distance. (ii) To study the effect of multivariate outliers on the mean and dispersion of quality scores of clustering configurations yielded by the determinantal consensus clustering. (iii) To adapt the determinantal consensus clustering to the case of very large datasets. The bottleneck of the method is the eigendecomposition of the kernel
matrix. This is a central step for obtaining an initial random subset of points with the determinantal point process. The computational complexity of the eigendecomposition of a \( n \times n \) symmetric matrix is \( O \left( n^3 \right) \). As \( n \) grows larger, the computation of the matrix spectral decomposition becomes expensive. We explore approximative ways to overcome this challenge with sparse matrix approximation to the kernel matrix. This is the topic we cover in a sequel paper on determinantal consensus clustering on large datasets.

**Acknowledgements**  This work was partially funded by a Discovery grant 2019-05444 from the Natural Sciences and Engineering Research Council of Canada, and by a Fundamental Research Projects grant PRF-2017-20 from the Institute for Data Valorization (IVADO) of Quebec, Canada.

8 Appendix: Proof that an increasing sequence of thresholds gives rise to hierarchical cluster partitions

Consider a given sequence of increasing thresholds \( \theta_1 < \theta_2 < \cdots < \theta_m \). As mentioned in Sect. 4, the associated consolidated matrices \( C^{(1)}, \ldots, C^{(m)} \) are nested in the sense that when \( s < t \), \( C^{(s)}_{ij} \geq C^{(t)}_{ij} \) for all pairs \( (i, j) \) \( s, t = 1, \ldots, m \). Let \( P_s \) and \( P_t \) be the connected component partitions associated with the consolidated matrices \( C^{(s)} \) and \( C^{(t)} \), respectively. Let points \( x_i \) and \( x_j \) be in the same connected component of \( P_t \). That is, there is a path \( x_i \leftrightarrow x_{l_1} \leftrightarrow \cdots \leftrightarrow x_{l_{g-1}} \leftrightarrow x_j \) of points satisfying \( C^{(t)}_{l_h,l_{h+1}} = 1, h = 0, \ldots, g - 1 \), with \( l_0 = i \), and \( l_g = j \). Since \( C^{(s)}_{ij} \geq C^{(t)}_{ij} \) for all pairs \( (i, j) \), necessarily points \( x_i \) and \( x_j \) are also in the same connected component of \( P_s \). The converse is not true. So suppose that points \( x_i \) and \( x_j \) are in the same connected component, say \( C_\ell \) of \( P_s \). That is, there is a path \( x_i \leftrightarrow x_{l_1} \leftrightarrow \cdots \leftrightarrow x_{l_{g-1}} \leftrightarrow x_j \) of points satisfying \( C^{(s)}_{l_h,l_{h+1}} = 1, h = 0, \ldots, g - 1 \), with \( l_0 = i \), and \( l_g = j \). Suppose as well that for some \( h \in \{0, \ldots, g - 1\} \), \( C^{(t)}_{l_h,l_{h+1}} = 0 \). Then the proportion of times the pair \( (x_i, x_j) \) appear together is at least \( \theta_s \), but less than \( \theta_t \). Points \( x_i \) and \( x_j \) are necessarily in different connected components of \( P_t \). This means that the component \( C_\ell \in P_s \) has split in several connected components isolated from all other components of \( P_s \). In this sense the sequence of thresholds \( \theta_1 < \theta_2 < \cdots < \theta_m \), leads to a sequence of partitions satisfying \( P_1 \supseteq P_2 \supseteq \cdots \supseteq P_m \), where \( P_s \supseteq P_t \) means that each connected component in \( P_t \) arises from a connected component of \( P_s \), and the number of connected components of \( P_t \) is larger or equal to the number of connected components of \( P_s \).

9 Appendix: Scattering formulas in the transformed space: using the kernel trick

Consider the RBF kernel \( \kappa(x_i, x_j) \) defined by (3). The mean scattering induced by the kernel on the data (Vert et al. 2004; Fan et al. 2010) is defined as
This formula is derived using the kernel trick which links kernel evaluations with inner products between transformed points in a high-dimensional space. It can be shown that this quantity corresponds to the average distance between the transformed points and the mean of the transformed points. Let \( V = \{ V_1, \ldots, V_K \} \) be a clustering configuration with \( K \) clusters. Also, let \( n_k \) be the size of cluster \( V_k \), and let \( J_k = \{ i : x_i \in V_k \}, k = 1, \ldots, K. \) As with the mean scattering, Fan et al. (2010) define similarly, the kernel induced scattering within each cluster \( V_k \),

\[
W_{V_k} = \frac{1}{n_k} \sum_{i \in J_k} \left\{ \kappa(x_i, x_i) - \sum_{j \in J_k} \frac{2 \kappa(x_i, x_j)}{n_k} + \sum_{j, \ell \in J_k} \frac{\kappa(x_j, x_\ell)}{n_k^2} \right\}^{\frac{1}{2}}
\]

which can be seen as the average distance between the mean of the clusters \( V_k \) in the transformed space and the transformed points of cluster \( V_k \). Finally, to measure the total scattering between clusters, one considers the distance between clusters in the transformed space,

\[
B^2(V_i, V_j) = \left| B(V_i, V_j) \right|^2 = \sum_{i, j \in J_i} \frac{\kappa(x_i, x_j)}{n_i^2} - \sum_{i \in J_i} \sum_{j \in J_j} \frac{2 \kappa(x_i, x_j)}{n_i n_j} + \sum_{j \in J_j} \frac{\kappa(x_j, x_j)}{n_j^2}.
\]

10 Appendix: Scheme to sample \( Y \sim DPP_S(L) \)

Below \( e_i \) represents the \( i \)th standard basis vector, a \( n \)-dimensional vector that is all zeros except for a one in the \( i \)th position. The span of a given finite collection of vectors \( V \), denoted by \( \text{span}(V) \), is the set containing of all linear combinations of the vectors in \( V \), i.e., \( \text{span}(V) = \{ \sum_{v \in V} a_v v \mid a_v \in \mathbb{R} \} \).

11 Appendix: Sensitivity analysis on kernel choice

As mentioned in Sect. 5.3, we considered three kernels: the linear kernel, \( \kappa(x_i, x_j) = x_i^T x_j \), and two variants of the \( q \)-stable kernel (Bilodeau and Nangue 2017), also known as the \( q \)-Gaussian kernel (Verleysen and François 2005), which is given by \( \kappa(x_i, x_j) = \exp \left( - \| x_i - x_j \|^q / \sigma^q \right), q \in (0, 2] \), where \( \sigma > 0 \) is the kernel bandwidth. The variants considered are the Laplace kernel \( (q = 1) \), and the Gaussian kernel \( (q = 2) \).

Two estimates of the bandwidth associated with each one of the two \( q \)-stable kernels studied were considered. For the Laplace kernel, the bandwidth estimates were the mean \( \hat{\sigma}^2 \) and the median \( \tilde{\sigma}^2 \) of all pairwise Euclidean distances. For the Gaussian kernel, the estimates were the mean \( \hat{\sigma}^2 \) and the median \( \tilde{\sigma}^2 \) of all pairwise squared Euclidean

\[
\text{DPP}_S(L)
\]
Algorithm 2: Sampling from a DPP

Input: eigendecomposition \( L = \sum_{i=1}^{n} \lambda_i(L) v_i v_i^T \)

Draw \((Z_1, \ldots, Z_n)\), where \( Z_i \sim \text{Bernoulli} \left( \frac{\lambda_i(L)}{\lambda_i(L) + 1} \right) \), \( i = 1, \ldots, n \)

\[ J \leftarrow \emptyset \]

for \( i = 1 \) to \( n \) do
  if \( Z_i = 1 \) then
    \( J \leftarrow J \cup \{i\} \)
  end

\[ V_J \leftarrow \{v_i : i \in J\} \]

\( Y \leftarrow \emptyset \)

while \( \text{card}(V_J) > 0 \) do
  Select \( i \) from \( \{1, 2, \ldots, n\} \) with probability \( \frac{1}{\text{card}(V_J)} \sum_{v \in V_J} \left( v^T e_i \right)^2 \)
  \( Y \leftarrow Y \cup \{i\} \)
  \( V_J \leftarrow \text{orthonormal basis for the subspace of span}(V_J) \) orthogonal to \( e_i \)
end

Output: \( Y \)

distances. Note that the Gaussian kernel with bandwidth estimate \( \hat{\sigma}^2 \) corresponds to our proposed DPP kernel.

We further investigated the behavior of the kernels on data whose clusters are non-Gaussian. For this simulation, we followed the setup of Sect. 5.2. Recall that the data were generated from 24 experimental conditions of a 3^3-factorial design. For this study, we generated five replicas from each experimental condition. The non-ellipsoidal (non-Gaussian) clusters were obtained by applying the inverse Box–Cox transformation to the datasets. As before, we selected \( \lambda = 1/3, \lambda = 1/2, \) and \( \lambda = 2 \)
Determinantal consensus clustering for the power parameter $\lambda$ of the Box–Cox transformations. We run the consensus DPP algorithm described in Sect. 4 on each dataset with the five different kernels. The values of the consensus DPP parameters were set to $a = 0.5$, $R = 200$, and $\tau = 0.6$. To select the optimal clustering configuration, we used the kernel-based validation index $KVI_L$ given by (6). To measure its quality, we use the adjusted Rand index (see Sect. 4.2). The results are displayed in Fig. 5 in the form of barplots. Each bar represents a different kernel or consensus method. The segments on top of the bars represent $\pm$ one standard error. We also compare the results from the different kernels with those yielded by CC-PAM and CC-$k$-means++.

For both $q$-stable kernels, estimating the bandwidth parameter with the median or the mean of all the pairwise distances do not produce any major difference in the ARI results.

References

Ankerst M, Breunig MM, Kriegel HP, Sander J (1999) Optics: ordering points to identify the clustering structure. ACM SIGMOD Rec 28(2):49–60

Ao SI, Yip K, Ng M, Cheung D, Fong P, Melhado I, Sham P (2005) Clustag: hierarchical clustering and graph methods for selecting tag SNPS. Bioinformatics 21:1735–6. https://doi.org/10.1093/bioinformatics/bti201

Arthur D, Vassilvitskii S (2007) K-means++: The advantages of careful seeding. In: Proceedings of the eighteenth annual ACM-SIAM symposium on discrete algorithms, USA, SODA ’07. Society for Industrial and Applied Mathematics, pp 1027–1035

Aurenhammer F (1991) Voronoi diagrams—a survey of a fundamental geometric data structure. ACM Comput Surv 23(3):345–405. https://doi.org/10.1145/116873.116880

Banfield JD, Raftery AE (1993) Model-based Gaussian and non-Gaussian clustering. Biometrics 49(3):803–821

Ben Hough J, Krishnapur M, Peres Y, Virág B (2006) Determinantal processes and independence. Probab Surv [electronic only] 3:206–229

Bicego M, Baldo S (2016) Properties of the Box–Cox transformation for pattern classification. Neurocomputing 218:390–400

Bien J, Tibshirani R (2011) Hierarchical clustering with prototypes via minimax linkage. J Am Stat Assoc 106:1075–1084. https://doi.org/10.1198/jasa.2011.tm10183

Bilodeau M, Nangue AG (2017) Tests of mutual or serial independence of random vectors with applications. J Mach Learn Res 18(1):2518–2557

Blatt M, Wiseman S, Domany E (1996) Superparamagnetic clustering of data. Phys Rev Lett 76:3251–3254

Blatt M, Wiseman S, Domany E (1997) Data clustering using a model granular magnet. Neural Comput 9(8):1805–1842

Borodin A, Olshanski G (2000) Distributions on partitions, point processes, and the hypergeometric kernel. Commun Math Phys 211:335–358. https://doi.org/10.1007/s002200050815 arXiv:math/9904010

Box GEP, Cox DR (1964) An analysis of transformations. J R Stat Soc B 26:211–252

Budiaji W, Leisch F (2019) Simple k-medoids partitioning algorithm for mixed variable data. Algorithms 12(9):1805–1842

Capó M, Pérez A, Lozano JA (2017) An efficient approximation to the k-means clustering for massive data. Knowl-Based Syst 117:56–69

Celebi ME, Kingravi HA, Vela PA (2013) A comparative study of efficient initialization methods for the k-means clustering algorithm. Expert Syst Appl 40(1):200–210. https://doi.org/10.1016/j.eswa.2012.07.021

Chaudhuri A, Kakde D, Sadek C, Gonzalez L, Kong S (2017) The mean and median criteria for kernel bandwidth selection for support vector data description. In: 2017 IEEE international conference on data mining workshops (ICDMW). IEEE, pp 842–849

Chen D, Xing K, Henson D, Sheng L, Schwartz AM, Cheng X (2009) Developing prognostic systems of cancer patients by ensemble clustering. J Biomed Biotechnol 2009:632786
Dua D, Graff C (2017) UCI machine learning repository. http://archive.ics.uci.edu/ml
Efron B, Tibshirani RJ (1994) An introduction to the bootstrap. CRC Press, Boca Raton
Ester M, Kriegel HP, Sander J, Xu X et al (1996) A density-based algorithm for discovering clusters in large spatial databases with noise. KDD 96:226–231
Fan Z, Jiang X, Xu B, Jiang Z (2010) An automatic index validity for clustering. In: Tan Y, Shi Y, Tan KC (eds) Advances in swarm intelligence. Springer, Berlin, pp 359–366
Filippone M, Camasstra F, Masulli F, Rovetta S (2008) A survey of kernel and spectral methods for clustering. Pattern Recogn 41(1):176–190
Florek K, Łukaszewicz J, Perkal J, Steinhaus H, Zubrzycki S (1951) Sur la liaison et la division des points d’un ensemble fini. Colloq Math 2:282–285
Forgy EW (1965) Cluster analysis of multivariate data: efficiency versus interpretability of classifications. Biometrics 21:768–769
Fraley C, Raftery AE (1998) How many clusters? Which clustering method? Answers via model-based cluster analysis. Comput J 41:578–588
Fränti P, Sieranoja S (2019) How much can k-means be improved by using better initialization and repeats? Pattern Recogn 93:95–112
Girolami M (2002) Mercer kernel-based clustering in feature space. IEEE Trans Neural Netw 13(3):780–784
Gonzalez TF (1985) Clustering to minimize the maximum intercluster distance. Theor Comput Sci 38:293–306
Hafiz Affandi R, Fox EB, Taskar B (2013) Approximate inference in continuous determinantal point processes. ArXiv e-prints arXiv:1311.2971
Hafiz Affandi R, Fox EB, Adams RP, Taskar B (2014) Learning the parameters of determinantal point process kernels. ArXiv e-prints arXiv:1402.4862
Han J, Kamber M, Pei J (2011) Data mining: concepts and techniques, 3rd edn. Morgan Kaufmann Publishers Inc., San Francisco
Hennig C (2019) Cluster validation by measurement of clustering characteristics relevant to the user, Chap 1. Wiley, Hoboken, pp 1–24
Herbrich R (2001) Learning kernel classifiers: theory and algorithms. MIT Press, Cambridge
Hinneburg A, Keim DA (1999) Optimal grid-clustering: towards breaking the curse of dimensionality in high-dimensional clustering. In: 25th International conference on very large databases, pp 506–517
Howley T, Madden MG (2006) An evolutionary approach to automatic kernel construction. In: Kollias S, Stafylopatis A, Duch W, Oja E (eds) Artificial Neural Networks—ICANN 2006. Springer, Berlin, pp 417–426
Hubert L, Arabie P (1985) Comparing partitions. J Classif 2(1):193–218. https://doi.org/10.1007/BF01908075
Ibrahim LF, Harbi MHA (2013) Using modified partitioning around medoids clustering technique in mobile network planning. arXiv preprint arXiv:1302.6602
Jain A, Dubes R (1988) Algorithms for clustering data. Prentice-Hall Inc, Upper Saddle River
Kang B (2013) Fast determinantal point process sampling with application to clustering. In: Burges C, Bottou L, Welling M, Ghahramani Z, Weinberger K (eds) Advances in neural information processing systems, vol 26. Curran Associates, Inc., New York, pp 2319–2327
Karatzoglou A, Smola A, Hornik K, Zeileis A (2004) kernlab—an s4 package for kernel methods in r. J Stat Softw 11(9):1–20
Katsavounidis I, Kuo CCJ, Zhang Z (1994) A new initialization technique for generalized Lloyd iteration. IEEE Signal Process Lett 1(10):144–146
Kaufmann L, Rousseseu P (1987) Clustering by means of medoids. In: Proceedings of the statistical data analysis based on the L1 norm conference, Neuchatel, Switzerland, vol 31, pp 405–416
Kulesza A, Taskar B (2012) Determinantal point processes for machine learning. Found Trends Mach Learn 5(2–3):123–286. https://doi.org/10.1561/2200000044
Lago-Fernández LF, Corbacho F (2010) Normality-based validation for crisp clustering. Pattern Recogn 43(3):782–795
Lewis D (1997) Reuters-21578 text categorization collection distribution 1.0. http://kdd.ics.uci.edu/databases/reuters21578/reuters21578.html
Lin D (1998) An information-theoretic definition of similarity. In: Proceedings of the 15th international conference on machine learning. Morgan Kaufmann, San Francisco, pp 296–304
Lloyd SP (1982) Least squares quantization in PCM. IEEE Trans Inf Theory 28:129–136

Springer
Maitra R (2009) Initializing partition-optimization algorithms. IEEE/ACM Trans Comput Biol Bioinf 6(1):144–157
Melnikov V, Chen WC, Maitra R (2012) Mixsim: an R package for simulating data to study performance of clustering algorithms. J Stat Softw 51(12):1–25. https://doi.org/10.18637/jss.v051.i12
Monti S, Tamayo P, Mesirov J, Golub T (2003) Consensus clustering: a resampling-based method for class discovery and visualization of gene expression microarray data. Mach Learn 52(1):91–118
Muñoz J, Murua A (2018) Building cancer prognosis systems with survival function clusters. Stat Anal Data Min ASA Data Sci J 11(3):98–110. https://doi.org/10.1002/sam.11373
Murua A, Wicker N (2014) The conditional-Potts clustering model. J Comput Graph Stat 23(3):717–739
Murua A, Stanberry L, Stuetzle W (2008) On Potts model clustering, kernel k-means, and density estimation. J Comput Graph Stat 17(3):629–658
Ng A, Jordan M, Weiss Y (2001) On spectral clustering: analysis and an algorithm. Adv Neural Inf Process Syst 14:849–856
Okabe A, Boots B, Sugihara K, Chiu SN (2000) Spatial Tessellations: Concepts and Applications of Voronoi Diagrams, 2nd edn. Series in Probability and Statistics. Wiley, Hoboken
Park HS, Jun CH (2009) A simple and fast algorithm for k-medoids clustering. Expert Syst Appl 36(2):3336–3341
Pena JM, Lozano JA, Larrañaga P (1999) An empirical comparison of four initialization methods for the k-means algorithm. Pattern Recogn Lett 20(10):1027–1040
Rand WM (1971) Objective criteria for the evaluation of clustering methods. J Am Stat Assoc 66(336):846–850
Saad D (1998) Online algorithms and stochastic approximations. Online Learn 5:3–6
Schölkopf B, Smola A, Müller KR (1998) Nonlinear component analysis as a kernel eigenvalue problem. Neural Comput 10(5):1299–1319
Schölkopf B, Tsuda K, Vert JP (2004) Kernel methods in computational biology. MIT Press, Cambridge
Schubert E, Roussieeuw PJ (2019) Faster k-medoids clustering: improving the PAM, CLARA, and CLARANS algorithms. In: International conference on similarity search and applications. Springer, pp 171–187
Sejdinovic D, Sriperumbudur B, Gretton A, Fukumizu K (2013) Equivalence of distance-based and RKHS-based statistics in hypothesis testing. Ann Stat 41:2263–2291
Shi J, Malik J (2000) Normalized cuts and image segmentation. IEEE Trans Pattern Anal Mach Intell 22(8):888–905
Smyth P (1997) Clustering sequences with hidden Markov models. In: Proceedings of the 9th international conference on neural information processing systems (NIPS 1996). MIT Press, Cambridge, pp 648–654
Strehl A, Ghosh J (2002) Cluster ensembles—a knowledge reuse framework for combining multiple partitions. J Mach Learn Res 3:583–617. https://doi.org/10.1162/153244303321897735
Stuetzle W (2003) Estimating the cluster tree of a density by analyzing the minimal spanning tree of a sample. J Classif 20(1):25–47
Stuetzle W, Nugent R (2010) A generalized single linkage method for estimating the cluster tree of a density. J Comput Graph Stat 19(2):397–418
Thygesen HH, Zwinderman AH (2004) Comparing transformation methods for DNA microarray data. BMC Bioinform 5(1):77
Vanschoren J, van Rijn JN, Bischl B, Torgo L (2013) Openml: networked science in machine learning. SIGKDD Explor 15(2):49–60. https://doi.org/10.1145/264190.2641198
Vapnik VN (1995) The nature of statistical learning theory. Springer, Berlin
Vega-Pons S, Ruiz-Shulcloper J (2011) A survey of clustering ensemble algorithms. Int J Pattern Recogn Artif Intell 25(03):337–372. https://doi.org/10.1142/S0218001411008683
Verleysen M, François D (2005) The curse of dimensionality in data mining and time series prediction. In: International work-conference on artificial neural networks. Springer, pp 758–770
Vert JP, Tsuda K, Schölkopf B (2004) A primer on kernel methods. Kernel Methods Comput Biol 47:35–70
Wang F, Landau DP (2001) Efficient, multiple-range random walk algorithm to calculate the density of states. Phys Rev Lett 86(10):2050
Wang W, Yang J, Muntz R et al (1997) Sting: a statistical information grid approach to spatial data mining. VLDB 97:186–195
Ward JH Jr (1963) Hierarchical grouping to optimize an objective function. J Am Stat Assoc 58(301):236–244
Watkins C (1999) Dynamic alignment kernels. In: Smola AJ, Bartlett PL, Schölkopf B, Schuurmans D (eds) Advances in large margin classifiers. MIT Press, pp 39–50

Xuan L, Zhigang C, Fan Y (2013) Exploring of clustering algorithm on class-imbalanced data. In: 8th International conference on computer science and education, ICCSE 2013, pp 89–93. https://doi.org/10.1109/ICCSE.2013.6553890

Yeung KY, Fraley C, Murua A, Raftery AE, Ruzzo WL (2001) Model-based clustering and data transformations for gene expression data. Bioinformatics 17:977–987

Publisher’s Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.