An Aposterioral Clusterability Criterion for k -Means++ and Simplicity of Clustering

Mieczysław A. Kłopotek¹

Received: 30 December 2019 / Accepted: 20 February 2020 / Published online: 12 March 2020
© The Author(s) 2020

Abstract

In this paper, the notion of a well-clusterable data set is defined combining the point of view of the objective of k -means clustering algorithm (minimizing the centric spread of data elements) and common sense (clusters shall be separated by gaps). Conditions are identified under which the optimum of k -means objective coincides with a clustering under which the data is separated by predefined gaps. Two cases are investigated: when the whole clusters are separated by some gap and when only the cores of the clusters meet some separation condition. A major obstacle for using known clusterability criteria is their reference to the optimal clustering which is NP hard to identify. In this paper, this obstacle is overcome. Compared to other approaches to clusterability, the novelty consists in the possibility of an a posteriori (after running k -means) check if the data set is well-clusterable or not. As the k -means algorithm applied for this purpose has polynomial complexity so does therefore the appropriate check. Additionally, if k -means++ fails to identify a clustering that meets clusterability criteria, with high probability the data is not well-clusterable.

Keywords k -means · Clusterability · Well-clusterable data · Gaps between clusters · Aposterior clusterability test

Introduction

Clusterability: The Intuition and the Failed Formalizations

It is a commonly observed phenomenon that the optimality criteria of most practically used clustering algorithms (like k -means) have a high theoretical computational complexity (are NP-hard). However, these algorithms perform quite well (converge quickly enough) yielding usable output in many (though not all) practical application cases. Apparently, the data must have the property that some data sets are better clusterable than other.

Though a number of attempts have been made to capture formally the intuition behind clusterability, none of these efforts seems to have been successful, as Ben-David exhibits in [10] in depth. He points at three important shortcomings of current state-of-the-art research results: the clusterability cannot be checked prior to applying a potentially NP-hard clustering algorithm to the data, known clusterability criteria impose

strong (impractical) separation constraints and the research nearly does not address popular algorithms. A recent paper by Ackerman et al. [1] partially eliminates some of these problems, but regrettably at the expense of introducing user-defined parameters that do not seem to be intuitive (in terms of one's imagination about what well-clusterable data are).

What Shall be Excluded from/Included into a Clusterability Criterion

Therefore, a different approach to defining what well-clusterable data are attempted in this paper. As Ben-David mentioned, the research in the area does not address popular algorithms except for ϵ -separatedness clusterability criterion related to k -means proposed by Ostrovsky et al. [20]. This paper is intended to contribute to applicability of clusterability criteria in that, following Ostrovsky's example, k -means is considered, and in particular its special version called k -means++, as in fact Ostrovsky did.¹

✉ Mieczysław A. Kłopotek
klopotek@ipipan.waw.pl

¹ Institute of Computer Science, Polish Academy of Sciences,
ul. Jana Kazimierza 5, 01-248 Warsaw, Poland

¹ Ostrovsky's algorithm differs from k -means++ in the initialization step of the first two seeds. k -means++ chooses the first seed randomly, and the subsequent seeds are chosen promoting the more distant ones (allocating to them higher probabilities of selection), whereas Ostrovsky selects carefully already the first two seeds. Then both algorithms proceed in the same way.

Fig. 1 Illustration of a special case where Ackerman’s method [1] would falsely recognize clustering structure in the data **a** the data, **b** the histogram of pair-wise distances—two modes visible

Furthermore, Ben-David expressed the concern that it is not a solution to the problem if the NP hardness is shifted from the data clustering algorithm to the data clusterability checking algorithm, because the problem becomes even worse. This paper addresses this concern. Last not least, the issue of the impractical gaps imposed by the clusterability criteria in the literature has been raised. Ben-David argues in [10] (see his “[Smaller Gaps Between Clusters](#)” section) that apparently the efforts in clusterability research on finding support for the hypothesis of “clustering is hard only if the clustering does not matter” have failed, mainly due to the fact that gaps between clusters that are required are too large for practical applications, as the popular algorithms behave reasonably even with significantly smaller gaps. But a closer look at the k -means objective shows (see “[Failure of Gaps as a Criterion for Clusterability](#)” section) that it does not make sense to build a clusterability criterion solely on the grounds of the gaps because k -means criterion does not rely solely on gaps between clusters, but also on their cardinalities as well as on their internal spread, which cannot be known in advance. If cardinalities are changed, k -means may prefer to split larger clusters and merge parts of them with smaller, though clearly separated clusters. If the k -means criterion shall coincide with separation of clusters, the gaps need to be large. Therefore, instead of seeking smaller gaps, this research concentrates on redefining the goal of clusterability research efforts. So it is proposed here to change the perspective. Instead of (or in addition to) seeking conditions for easiness of clustering of a given data set, one shall look for definition of a data set (data set generator) for which optimal clustering is known in

advance and an algorithm returns the ground truth nearly for sure. This change of perspective will lead to a practical application of clusterability concept consisting in testing the algorithm behaviour under varying degrees of violating the clusterability conditions.

Having been freed from the need to seek the smallest possible gaps, one can also weaken the problems with NP hardness of stating the clusterability of the data set. In particular, one shall not require that it has to be said beforehand (before clustering) whether or not the data is well-clusterable. Instead one shall require that one is able to state aposteriorically *whether or not* the data is well-clusterable according to well-clusterability criteria that were assumed, in polynomial time. Note that this is a tremendous progress over clusterability criteria defined so far. None of the clusterability criteria discussed by Ben-David [10] fits this requirement and the criterion proposed in [1] can be shown to be invalid for simple data sets (see “[Failure of Multi-modality Detection for Detection of Gaps Between Data](#)” section, Fig. 1). This paper resolves in this way a serious bottleneck in the clusterability research. The issue of measuring the deviation from well-separatedness is not covered here. Nonetheless, the presented clusterability criterion opens a way to attack this issue also, because the criterion is verifiable ex post and the data can be checked for clusterability by at least one popular algorithm.

Clusterability Challenge in the k -Means Algorithm Family

This study is restricted to the k -means family of algorithms. The restriction is not too serious as the algorithms of this

family are broadly used and there exist quite large number of variants, starting with early work of Lloyd, Forgy, MacQueen and Hartigan–Wong, to k -means++, spherical k -means, their fuzzified version, kernel variants, k -harmonic means, and many other.² k -means is used also as a part of other algorithms, like those of spectral analysis [23].

While studying clusterability, we concentrate on the k -means++ algorithm's initialization phase, introduced by Arthur and Vassilvitskii [5]. It is a well-known problem with k -means basic version (with random initialization) that it tends to stick in local minima if the initial seeding is not very good. Ways to overcome this shortcoming have been sought, like the mentioned k -means++ algorithm that ensures a better seeding. However, other approaches like k -harmonic-means by Zhang et al. [26], power- k -means by Xu et al. [25] and entropy-regularized-power- k -means by Chakraborty et al. [13] that attempt to reduce the dependence of the performance on the initialization. Their advantage is that they smoothen the landscape of the optimized function so that the chance of sticking into local optima is reduced. These approaches are, however, not suitable for our purposes. The entropy-regularized-power- k -means has a target function (quality function or optimization criterion) that is definitely different from that of k -means, though many interesting theoretical properties like strong consistency have been shown. These properties do not warrant however reaching their global optimum. The k -harmonic-means target function differs also from that of k -means though it may be used as an initial phase of the algorithm (instead of a clever seeding) followed by proper k -means, and it was shown empirically that better target function values are found. Only power- k -means has the target function that agrees with that of k -means. However, its complex mathematical form makes it difficult to derive a clusterability criterion that would be superior to that proposed here for k -means++, though this would be worth investigation. Nonetheless, any data set clusterable for k -means++ should also turn out to be clusterable for power- k -means as it is less sensible to bad initialization (that can still happen with k -means++).

The following challenges have to be faced within the k -means family according to Ben-David [10]:

- the clusterability criteria in the literature (e.g. Ostrovsky et al. [20]) refer to the optimal cost function value of k -means [see Eq. (1)]—but the actual value of this optimal solution is not known
- people are accustomed to associate well-clusterable data with ones of large gaps between clusters—but the optimal cost function of k -means is also influenced by cluster

sizes, so that the gap sufficient for one set of clusters will prove insufficient for some other (see “[Failure of Gaps as a Criterion for Clusterability](#)” section)

- the cost function of k -means usually has multiple local minima and the real-world k -means algorithms usually tend to stick at some local minimum (see e.g. [24, Chapter 3]).

For these reasons, when comparing the results of various k -means brands on real data, it is hard to distil the reason why their results differ: is it because the data are not clusterable or that the cost function optimum does not agree with common sense split into well-separated clusters or the algorithm is unable to discover the optimal clustering (systematically misses it).

Therefore, a clusterability criterion is sought in this paper such that:

- it is based on the gap size between clusters and other cluster characteristics, that can be computed by inspection of an obtained clustering (not referring to the optimal one),
- if the clustering obtained meets the clusterability criteria, then this is the real optimal clustering,
- if a special algorithm (k -means++ is meant here) fails to find a clustering meeting the clusterability criteria, then with high probability the data is not well-clusterable at all by any algorithm,
- there exists the possibility to generate a data set matching the clusterability conditions for various constellations of the cluster sizes (cardinality, spread), dimensionalities, number of clusters etc.

Given such a tool at disposal, one can investigate algorithm's capability to find the optimal clustering in the easy case, compare the algorithms in their performance in an easy case, and then compare their relative performance when the clusterability property degenerates, for example via decreasing the size of the gap between clusters.

Our Contribution

This research is confined to providing the tool in terms of the new clusterability criterion, and only a small demonstration is made, how the degenerative behaviour of algorithms may be studied. Our contribution encompasses:

- Two brands of well-clusterability criteria for data to be clustered via k -means algorithm, that can be verified ex post (both positively and negatively) without great computational burden [inequalities (2) and (3) in “[Our Basic Approach to Clusterability](#)” section, and inequalities (15) and (16) in “[Smaller Gaps Between Clusters](#)” section].

² See e.g. [24, Chapter 3] for a review of the k -means algorithm family.

- Demonstration, that the structure of well-clusterable data (according to these criteria) is easy to recover [see Theorems 1(i) and 5(i)].
- Demonstration that if well-clusterable data structure (in that sense) was not discovered by k -means++, then there is no such structure in the data [with high probability—see Theorems 1(ii) and 5(ii)].
- Demonstration that large gaps between data clusters are not sufficient to ensure well-clusterability by k -means (see “[Failure of Gaps as a Criterion for Clusterability](#)” section).

Paper Structure

The structure of this paper is as follows: in “[The Problem of Clusterability in the Previous Work](#)” section, the previous work on the topic of clusterability is recalled and a brief introduction to the k -means algorithm and its special case k -means++ is given. “[Failure of Gaps as a Criterion for Clusterability](#)” section shows that large gaps are not sufficient for well-clusterability. “[Our Basic Approach to Clusterability](#)” section introduces the first version of well-clusterability concept and shows that data well-clustered in this sense are easily learnable via k -means++. This concept has the drawback that no data points (outliers) can lie in wide areas between the clusters. Therefore “[Core-Based Approach to Clusterability](#)” section proposes a core-based well-clusterability concept and shows that data well-clustered in this sense are also easily learnable via k -means++. The concept of cluster core itself is introduced and investigated in “[Smaller Gaps Between Clusters](#)” section and a method determining proper gap size under these new conditions is derived in “[Core-Based Global \$k\$ -Means Minimum](#)” section. In “[Experimental Results](#)” section, some experimental results are reported concerning performance of various brands of k -means algorithms for data fulfilling the clusterability criteria proposed in this paper. “[Discussion](#)” section contains a brief comparison of our clusterability criteria with those discussed by Ben-David [10]. In “[Conclusion](#)” section some conclusions are drawn.

The Problem of Clusterability in the Previous Work

Clusterability Property: Easy to Use, Hard to Verify

Intuitively the clusterability, as suggested by Ackerman et al. [2], shall be a function taking a set of points and returning a real value saying how “strong” or “conclusive” is the clustering structure of the data. This intuition, however, turns out not to be formalized in a uniform way. A large number of formal definitions have been proposed. Ackerman and

Ben-David [2] studied several of them. They concluded that two phenomena co-occur across the various formalizations: on the one hand, well-clusterable data sets (with high “clusterability” value) are computationally easy to cluster (in polynomial time), but on the other hand identification whether or not the data is well-clusterable is NP-hard.

Major Brands of Clusterability Criteria and Their Shortcomings

Ben-David [10] performed an investigation of the concepts of clusterability from the point of view of the capability of “not too complex” algorithms to discover the cluster structure. He verified negatively the working hypothesis that “Clustering is difficult only when it does not matter” (the *CDNM* thesis).

He considered the following notions of clusterability, present in the literature:

- *Perturbation Robustness* meaning that small perturbations of distances/positions in space of set elements do not result in a change of the optimal clustering for that data set. Two brands may be distinguished: additive, proposed by Ackerman and Ben-David [2], and multiplicative ones, proposed by Bilu and Linial [12] (the limit of perturbation is upper-bounded either by an absolute value or by a coefficient).
- *ϵ -Separatedness* meaning that the cost of optimal clustering into k clusters is less than ϵ^2 times the cost of optimal clustering into $k - 1$ clusters, by Ostrovsky et al. [20]—here an explicit reference to the k -means objective is made.
- *(c, ϵ) -Approximation-Stability* by Balcan et al. [8] meaning that if the cost function values of two partitions differ by the factor c , then the distance (in some space) between the partitions is at most ϵ . As Ben-David recalls, this implies the uniqueness of optimal solution.
- *α -Centre Stability* by Awasthi et al. [7] meaning, for any centric clustering, that the distance of an element to its cluster centre is α times smaller than the distance to any other cluster centre under optimal clustering.
- *$(1 + \alpha)$ Weak Deletion Stability* by Awasthi et al. [6] meaning that given an optimal cost function value OPT for k centric clusters, then the cost function of a clustering obtained by deleting one of the cluster centres and assigning elements of that cluster to one of the remaining clusters should be bigger than $(1 + \alpha) \cdot OPT$.

Under these notions of clusterability, algorithms have been developed clustering the data nearly optimally in polynomial times, when some constraints are matched by the mentioned parameters. However, these conditions seem to be rather extreme. Ben-David [10] points at concrete numerical

disadvantages. For example, given the (c, ϵ) -Approximation-Stability [8], polynomial time clustering requires that, in the optimal clustering (beside its uniqueness), all but an ϵ -fraction of the elements, are 20 times closer to their own cluster centre than to every other cluster centre. ϵ -Separatedness requires that the distance to its own cluster centre must be at least 200 times closer than to every other cluster element [20]. And this is still insufficient if the clusters are not balanced. A ratio of 10^7 is deemed by these authors as sufficient. $(1 + \alpha)$ Weak Deletion Stability [6] demands distances to other clusters being $\log(k)$ times the “average radius” of the own cluster. The perturbational stability [2] induces exponential dependence on the sample size.

Gap as a Component of Clusterability Criteria

One can draw a certain important conclusion from the concepts of clusterability mentioned above: people agree that a data set is well clusterable if each cluster is distant (widely separated) from the other clusters. This idea occurs in many other clusterability concepts. Eptner et al. [15] consider the data as clusterable when the minimum between-cluster separation exceeds the maximum in-cluster distance (called elsewhere “perfect separation”).³ Balcan et al. [9] propose to consider data as clusterable if each element is closer to all elements in its cluster than to all other data (called also “nice separation”).⁴ k -means reflects the Balcan concept “on average” that is each element average squared distance to elements of the same cluster is smaller than the minimum (over other clusters) averaged squared distance to elements of a different cluster. Kumar and Kannan [18], explicitly concentrating on k -means objective, define clusterability via a proximity condition as follows: Any point projected on a line connecting its own cluster centre and some other cluster centre should be closer to its own cluster centre by a “sufficiently large” gap depending on the number of clusters and inverted squared cluster cardinalities.

Kushagra et al. [19] consider clusterability from the point of view of a structure in the data. They allow for noise in the data, but insist that the noise does not create structures by itself. They refrain from optimizing a cost function. They show that discovery of clusters is not possible without assumption of structure in the data or without assumption of structureless noise. Ackerman et al. [3] consider the clusterability from the perspective of distortion of clusters

by malicious points. It turns out that from this perspective k -means performs better than various other algorithms. With respect to our research, they also insist that the proportions between cluster sizes play a significant role ensuring proper clustering.

Clusterability as a Function of Both Data and the Algorithm

Ackerman and Dasgupta [4] move the focus on clusterability from the clusterability as a property of the data alone to the pair of (data type, algorithm type). In that paper, they are interested in incremental algorithms only and show that an incremental version of k -means performs poorly under perfect and nice separation. In the same spirit Ben-David and Haghtala [11] investigated clusterability by k -centroidal algorithms (a class of algorithms including k -means) via robustifying an algorithm against noise in the data by either clustering the noise into separate clusters or cutting off too distant points. Tang [22] investigates a clusterability criterion for his own version of k -means, based on the requirement that the cluster centres are separated by some distance, which is dependent upon *ground truth optimal clustering*. Recently Ackerman et al. [1] derived a method for testing clusterability of data based on the large gap assumption. They investigate the histogram of (all) mutual dissimilarities between data points. If there is no data structure, the distribution should be unimodal. If there are distant clusters, then there will occur one mode for short distances (within clusters) and at least one for long distances (between clusters). Hence, to detect clusterability, they apply tests of multimodality, namely the Dip and Silverman tests, as suggested by Hartigan and Hartigan [16] and Silverman [21] resp.. But the criterion of a sufficiently large gap between clusters is not reflected in various clustering function objectives, for example k -means which may reach an optimum with poorly separated clusters in spite of the fact that there exists an alternative partition of data with a clear separation between clusters in the data, as demonstrated in “[Failure of Gaps as a Criterion for Clusterability](#)” section. It will be also demonstrated in “[Failure of Multi-modality Detection for Detection of Gaps Between Data](#)” section that multimodal distributions can be detected by Ackerman’s method even if there is no structure in the data.

Clusterability as Agreement Between Global and Local Clustering

Cohen-Addad [14] raises the claim that data are clusterable (in terms of various stability criteria) if the global clustering can be well approximated by local one. Our work can be perceived in this spirit in that we try to achieve coincidence

³ It has been shown in the literature that under this notion of well-clusterability single link algorithm can detect clusters separated in such a way. It has also been shown that centre-based algorithms like k -means may fail to detect such clusters, see e.g. Ackerman and Dasgupta [4].

⁴ It has been shown in the literature that this notion of well-clusterability is hard to decide in a data set, see e.g. [4].

of clusters based on separability with global cost function minimum.

Clusterability Criteria not Verifiable a Priori

Ben-David [10] raises a further important point: It is impossible to verify a priori if the data fulfils the clusterability criterion in practically all abovementioned methods (except [1], which has a flaw by itself, see “[Failure of Multi-modality Detection for Detection of Gaps Between Data](#)” section). The reason is that these criteria refer either to all possible clusterings or to optimal clustering so that one does not have the possibility to verify in polynomial time whether or not the data set is clusterable, before one starts clustering (but usually computing the optimum is NP-hard). In this paper, however, we stress that the situation is even worse. Even at the termination of the clustering algorithm, one is unable to say whether or not the clustered data set turned out to be well-clusterable. For example, the ϵ -Separatedness criterion requires that the nearly optimal solution for clustering into k and $k - 1$ elements is known. While one can usually get the upper approximations for the cost functions in both cases, one needs actually the lower approximation for $k - 1$ in order to decide ex post if the data was well-clusterable, and hence whether or not one can say that one approximated the correct solution in some way. But this lower approximation for $k - 1$ can only be obtained if $k - 1 = 1$. Therefore, the issue is not decidable for $k > 2$. Tang’s [22] criterion is certainly better, though also based on solution to optimality criterion, because one can sometimes decide ex post that the clusterability criterion was fulfilled. (the distance between clusters needs to be greater than a product of optimal clustering cost function and reversed squared roots of cluster cardinalities, which may be upper-bounded by the actual clustering cost function and the number 2.) But even if the algorithm actually finds optimal clustering, one cannot be sure that the solution is optimal even if the clusterability criterion is met.

The issue of ex post decision on clusterability seems nevertheless to be simpler to solve than the a priorical one; therefore, we will attack it in this paper. We are unaware that such an issue was even raised in the past. Though the criteria of [9, 15] can clearly be applied ex post to see if the clusterability criteria hold in the resulting clustering, these approaches fail to solve the inverse issue: what if the clusterability criteria are not matched by the result clustering—is the data non-clusterable? Could no other algorithm discover the clusterable structure? One shall note at this point that the approach in [1] is different with this respect. Compared to methods requiring finding the optimum first, Ackerman’s approach seems to fulfil Ben-David requirement, that one can see if there is clusterability in the data before starting the clustering process. Ackerman’s clusterability criterion can be computed in polynomial time because the computation

of the histogram of dissimilarities is quadratic in sample size. But at an in-depth-investigation, the Ackerman’s clusterability determination method misses one important point: it requires a user-defined parameter and the user may or may not make the right guess. Furthermore, even if clusterability is decided by Ackerman’s tests, it is still uncertain if k -means algorithm will be willing to find such a clustering that fits Ackerman’s clusterability criterion. Beside this, as visible in Fig. 1, one can easily find counterexamples to their concept of clusterability. The left image shows that there is a single cluster there, but the histogram to the right has two modes, indicating a two-cluster structure.

In summary, the issue of an aposteriorical determination if the data were clusterable, remains an open issue. Therefore, it seems to be justified to restrict oneself to a problem as simple as possible in order to show that the issue is solvable at all. In this paper, we will limit ourselves to the issue of clusterability for the purposes of k -means algorithm.⁵ Furthermore, we restrict ourselves to determine such cases when the clusterability is decidable “for sure”.

The first problem to solve seems to be to get rid of the dependence on the undecidedness of optimality of the obtained solution.

A Brief Introduction to the k -Means Family

Before proceeding let us recall the definition of k -means cost function $Q(\mathcal{C})$ for the clustering $\mathcal{C}$.

$$Q(\mathcal{C}) = \sum_{i=1}^m \sum_{j=1}^k u_{ij} \| \mathbf{x}_i - \boldsymbol{\mu}_j \|^2 = \sum_{j=1}^k \frac{1}{2n_j} \sum_{\mathbf{x}_i \in C_j} \sum_{\mathbf{x}_l \in C_j} \| \mathbf{x}_i - \mathbf{x}_l \|^2 \quad (1)$$

for a dataset $\mathbf{X}$ under some partition $\mathcal{C} = \{C_1, \dots, C_k\}$ into the predefined number k of clusters, $C_1 \cup \dots \cup C_k = \mathbf{X}$, where u_{ij} is an indicator of the membership of data point $\mathbf{x}_i$ in the cluster C_j having the centre at $\boldsymbol{\mu}_j = \frac{1}{|C_j|} \sum_{\mathbf{x} \in C_j} \mathbf{x}$.

The k -means algorithm starts with some initial guess of the positions of $\boldsymbol{\mu}_j$ for $j = 1, \dots, k$ and then alternating two steps: cluster assignment and centre update till some convergence criterion is reached, e.g. no changes in cluster membership. The cluster assignment step updates u_{ij} values so that each element $\mathbf{x}_i$ is assigned to a cluster represented by the closest $\boldsymbol{\mu}_j$. The centre update step uses the update formula $\boldsymbol{\mu}_j = \frac{1}{|C_j|} \sum_{\mathbf{x} \in C_j} \mathbf{x}$.

The k -means++ algorithm is a special case of k -means where the initial guess of cluster centres proceeds as follows. $\boldsymbol{\mu}_1$ is set to be a data point uniformly sampled from

⁵ The k -means algorithm seems to be quite popular in various variants both in traditional, kernel and spectral clustering. Hence the results may be still of sufficiently broad importance.

X. The subsequent cluster centres are data points picked from $\mathbf{X}$ with probability proportional to the squared distance to the closest cluster centre chosen so far. For details check the paper of Arthur and Vassilvitskii [5]. Note that the algorithm proposed by Ostrovsky et al. [20] differs from the k -means++ only by the non-uniform choice of the first cluster centre. (The first pair of cluster centres should be distant, and the choice of this pair is proportional in probability to the squared distances between data elements.)

Non-suitability of Gap-Based Clusterability Criteria for k -Means

Failure of Multi-modality Detection for Detection of Gaps Between Data

Let us discuss more closely the relationship between the gap-based well-clusterability concepts developed in the literature and the actual optimality criterion of k -means. Specifically let us consider the approaches to clusterability of [1, 7, 9, 15].

Intuitively, if the groups of data points occur in the data and there are large spaces between these groups, then it should be these groups that will be chosen as the actual clustering. On the other hand, if there are no gaps between the groups of data points, then one would expect that the data are not considered as well clusterable. Furthermore, if the data is well clusterable, one would expect a reasonable clustering algorithm to discover easily such a well-clusterable data structure.

However, these intuitions prove wrong in case of k -means. Let us first point to the fact that the method of [1] may indicate a clear bimodal structure in the data where there are no gaps in the data. We are unaware of anybody pointing at this weakness of well-clusterability in [1]: imagine a thin ring uniformly covered with data points (see Fig. 1a). One would be reluctant to say that there is a clustering structure in such data. Nonetheless, two obvious modes in such data are visible. The thinner the ring (the closer to a circle), the more obvious the reason for the multi-modality will be: one gets closer and closer to the following function. Consider the angle α centred at the centre of the circle (“thin ring”). To calculate distances between points, one can restrict himself to angles with measure $0^\circ \leq \alpha \leq 180^\circ$ (or $0 \leq \alpha \leq \pi$). The number of elements within the angle would be approximately proportional to this angle. The distance between cutting points of this angle on the circle, given a radius r of the circle, will amount to $x = 2r \sin \frac{\alpha}{2}$. Consequently $\alpha = 2 \arcsin \frac{x}{2r}$. To determine the density of distances, a derivative has to be computed

$$\frac{d\alpha}{dx} = \frac{1}{dx/d\alpha} = \frac{1}{d(2r \sin \frac{\alpha}{2})/d\alpha} = \frac{1}{r \cos \frac{\alpha}{2}} = \frac{1}{r \sqrt{1 - \sin^2 \frac{\alpha}{2}}} = \frac{1}{r \sqrt{1 - \frac{x^2}{4r^2}}}.$$

This function has a minimum at $x = r$ and grows towards both $x = 0$ and $x = 2r$. It is reflected in the histogram in Fig. 1b, where $r = 5$ is assumed. As the points occur not on a circle, but on a ring, more distances close to zero occur in the histogram.

Failure of Gaps as a Criterion for Clusterability

On the other hand, even if there are gaps between groups of data, for example those required by [7, 9, 15], k -means optimum may not lie in the partition exhibiting gap-based well-clusterability property in spite of its existence, and not only for these gaps, but also for any arbitrary many times larger ones. As [15] is concerned, it may be considered as a special case of [9]. Awasthi et al. [7] may be viewed in turn as a strengthening of the concept of [15]. Therefore, let us discuss a situation in which both perfect and nice separation criteria are identical that is of two clusters. It will be shown that whatever α is assumed in the α -stability concept, k -means fails to be optimal under unequal class cardinalities. Let these clusters, C_A, C_B be each enclosed in a ball of radius r and the distance between ball centres should be at least $4r$. It has been demonstrated in [17] that under these circumstances the clustering of data into C_A, C_B reflects a local minimum of k -means cost function. But it is not the global minimum, as will be shown subsequently. So the criteria of Epter et al. [15], Balcan et al. [9] and Awasthi et al. [7] cannot be viewed as realistic definitions of well-clusterability at least for k -means. Subsequently, whenever we say that a cluster is enclosed in a ball of radius r , we mean at the same time that the ball is centred at gravity centre of the cluster.

Assume, for purposes of demonstration, that both clusters are of different cardinalities n_A, n_B and let $n_A > n_B$. It will be shown that whatever distance between both clusters, one can get such a proportion of n_A/n_B that the clustering into C_A, C_B is not optimal. Let us consider a d -dimensional space. Let us select the dimension that contributes most to the variance in cluster C_A . So the variance along this direction amounts to at least the overall variance divided by d . Let us denote this variance component as V_d . Consider this coordinate axis responsible for V_d to have the origin at the cluster centre of C_A . Project all the points of cluster C_A on this axis. The variance of projected points will be just V_d . Split the projected data set into two parts P_1, P_3 , one with coordinate above 0 and the rest. Let P'_1, P'_3 be the original data points which were projected onto P_1, P_3 resp. Then $C_A = P'_1 \cup P'_3$ and $P'_1 \cap P'_3 = \emptyset$. Let the centres of P_1, P_3 lie x_1, x_3 away from the cluster centre. Let n_1 data points of P_1 be at most x_1 distant from the origin, and n_2 more than x_1 from the origin. Let there be n_3 data points of P_3 be at most x_3 distant from the origin, and n_4 more than x_3 from the origin. Obviously, $n_1 + n_2 + n_3 + n_4 = n_A, |P_1| = n_1 + n_2$,

$|P_3| = n_3 + n_4$. As zero is assumed to be the C_A cluster centre on this line, also $x_1 \cdot (n_1 + n_2) = x_3 \cdot (n_3 + n_4)$ holds. Furthermore, as the cluster is enclosed in a ball of radius r centred at its gravity centre, both $x_1 \leq r$ and $x_3 \leq r$. Under these circumstances, let us ask the question whether for a V_d some minimal values of x_1, x_3 are implied. Because if so, then by splitting the cluster C_A into P'_1, P'_3 and by increasing the cardinality of C_A , the split of data into $P'_1, P'_3 \cup C_B$ will deliver a lower Q value so that for sure the clustering into C_A, C_B will not be optimal.

Note that $V_d = (\text{Var}(P_1) + x_1^2) \cdot (n_1 + n_2) + (\text{Var}(P_3) + x_3^2) \cdot (n_3 + n_4) / n_A$. The n_1 points of P_1 closer to origin than x_1 are necessarily not more than x_1 distant from P_1 gravity centre. Therefore, the remaining n_2 points cannot be more distant than $x_1 \frac{n_1}{n_2}$. Hence $\text{Var}(P_1) \leq x_1^2 n_1 + \left(x_1 \frac{n_1}{n_2}\right)^2 n_2$. By analogy $\text{Var}(P_3) \leq x_3^2 n_3 + \left(x_3 \frac{n_3}{n_4}\right)^2 n_4$.

It is visible that

$$V_d \leq \frac{\frac{x_1^2(n_1+n_1^2/n_2+n_1+n_2)}{n_1+n_2} \cdot (n_1+n_2) + \frac{x_3^2(n_3+n_3^2/n_4+n_3+n_4)}{(n_3+n_4)} \cdot (n_3+n_4)}{n_A}$$

that is

$$V_d \leq \frac{x_1^2 \cdot (n_1 + n_1^2/n_2 + n_1 + n_2) + x_3^2 \cdot (n_3 + n_3^2/n_4 + n_3 + n_4)}{n_A}.$$

Note that one can delimit n_2, n_4 from below due to the relationship: $(r - x_1) \cdot n_2 \geq n_1 \cdot x_1, (r - x_3) \cdot n_4 \geq n_3 \cdot x_3$.

Therefore

$$V_d \leq \left(x_1^2 \cdot (n_1 + n_1^2 \cdot (r - x_1) / n_1 / x_1 + n_1 + n_2) + x_3^2 \cdot (n_3 + n_3^2 \cdot (r - x_3) / n_3 / x_3 + n_3 + n_4) \right) / n_A.$$

Hence

$$\begin{aligned} V_d &\leq \left(x_1^2 \cdot (2 \cdot n_1 + n_2) + n_1^2 \cdot (r - x_1) \cdot x_1 / n_1 \right. \\ &\quad \left. + x_3^2 \cdot (2 \cdot n_3 + n_4) + n_3^2 \cdot (r - x_3) \cdot x_3 / n_3 \right) / n_A \\ V_d &\leq \left(x_1^2 \cdot (2 \cdot n_1 + n_2) + n_1 \cdot (r - x_1) \cdot x_1 \right. \\ &\quad \left. + x_3^2 \cdot (2 \cdot n_3 + n_4) + n_3 \cdot (r - x_3) \cdot x_3 \right) / n_A \\ V_d &\leq \left(x_1^2 \cdot (n_1 + n_2) + n_1 \cdot r \cdot x_1 \right. \\ &\quad \left. + x_3^2 \cdot (n_3 + n_4) + n_3 \cdot r \cdot x_3 \right) / n_A. \end{aligned}$$

Recall that $x_1 \cdot (n_1 + n_2) = x_3 \cdot (n_3 + n_4)$. Therefore

$$V_d \leq \left(x_1^2 \cdot (n_1 + n_2) + n_1 \cdot r \cdot x_1 + (x_1 \cdot (n_1 + n_2) / (n_3 + n_4))^2 \cdot (n_3 + n_4) + n_3 \cdot r \cdot x_3 \right) / n_A$$

which is equivalent to

$$V_d \leq \left(x_1^2 \cdot (n_1 + n_2) + n_1 \cdot r \cdot x_1 + x_1^2 \cdot (n_1 + n_2)^2 / (n_3 + n_4) + n_3 \cdot r \cdot x_3 \right) / n_A$$

By rearranging the terms one gets:

$$V_d \leq \left(x_1^2 \cdot (n_1 + n_2) \cdot n_A / (n_3 + n_4) + n_1 \cdot r \cdot x_1 + n_3 \cdot r \cdot x_3 \right) / n_A.$$

Let us increase the right-hand side by adding to the nominator $n_2 \cdot r \cdot x_1 + n_4 \cdot r \cdot x_3$. This implies

$$V_d \leq \left(x_1^2 \cdot (n_1 + n_2) \cdot n_A / (n_3 + n_4) + (n_1 + n_2) \cdot r \cdot x_1 + (n_3 + n_4) \cdot r \cdot x_3 \right) / n_A.$$

Let us substitute $x_1 = \frac{x_3 \cdot (n_3 + n_4)}{n_1 + n_2}$. Then we obtain

$$V_d \leq \left((x_3 \cdot (n_3 + n_4) / (n_1 + n_2))^2 \cdot (n_1 + n_2) \cdot n_A / (n_3 + n_4) + 2 \cdot (n_3 + n_4) \cdot r \cdot x_3 \right) / n_A.$$

Hence

$$V_d \leq \left(x_3^2 (n_3 + n_4)^2 / (n_1 + n_2) \cdot n_A / (n_3 + n_4) + 2 \cdot (n_3 + n_4) \cdot r \cdot x_3 \right) / n_A.$$

One can delimit $n_1 + n_2$ from below due to relationship $x_3 \cdot (n_3 + n_4) = (n_1 + n_2) \cdot x_1 \leq (n_1 + n_2) \cdot r$ because $x_1 \leq r$. It implies that $\frac{1}{n_1 + n_2} \leq \frac{r}{x_3 \cdot (n_3 + n_4)}$. Therefore

$$V_d \leq \left(x_3^2 \cdot (n_3 + n_4)^2 \cdot r / x_3 / (n_3 + n_4) \cdot n_A / (n_3 + n_4) + 2 \cdot (n_3 + n_4) \cdot r \cdot x_3 \right) / n_A$$

which simplifies to

$$\begin{aligned} V_d &\leq (x_3 \cdot r \cdot n_A + 2 \cdot (n_3 + n_4) \cdot r \cdot x_3) / n_A \\ V_d &\leq x_3 \cdot r \cdot (n_A + 2 \cdot (n_3 + n_4)) / n_A \end{aligned}$$

Clearly $n_3 + n_4 < n_A$, which implies $V_d \leq 3 \cdot x_3 \cdot r$. This means that

$$x_3 \geq V_d / 3 / r$$

Now let us show that when scaling up n_A it pays off to split the first cluster and to attach the contents of the second one to one of the parts of the first. Let us increase the cardinality of C_A b times simply by replacing each data element by b data elements collocated at the same place in space. In this way, one keeps V_d when increasing $|C_A|$. So the sum of squared distances between centre and elements of the cluster C_A , $\text{SSC}(C_A)$ will be kept below $V_d \cdot d \cdot n_A b$ ($\text{SSC}(C_A) \leq V_d \cdot d \cdot n_A b$).

Let $n_1 + n_2$ be the minority among data points—then x_1 is larger and x_3 is smaller of the two, because of $x_1 \cdot (n_1 + n_2) = x_3 \cdot (n_3 + n_4)$. Let P'_1, P'_3 be the subsets of C_A yielding upon the aforementioned projection the mentioned sets P_1, P_3 . Then if one would split C_A into P'_1, P'_3 , the sum of squared distances to respective cluster centres of P'_1, P'_3 would decrease by at least $x_3^2 n_A b$, because $\text{SSC}(P_1 \cup P_3) - x_3^2 n_A b \geq \text{SSC}(P_1 \cup P_3) - x_1^2 (n_1 + n_2) b - x_3^2 (n_3 + n_4) b \geq \text{SSC}(P_1) + \text{SSC}(P_3)$, and the distances

Table 1 Dependence of the number of errors on M to m proportion under fixed gap $g/r_{\max} = 2$

$M, m =$	15, 9	480, 9	720, 9	960, 9
Hartigan–Wong 1 ×	0 ± 0	961.3 ± 8.8	974.3 ± 3.2	980.9 ± 4.9
1000 execs in s/relQ	0.32/1	0.57/1.07	0.67/1	0.79/1
Lloyd 1 ×	1.7 ± 3.6	962.5 ± 5.6	974.1 ± 4.9	980.4 ± 4.5
1000 execs in s/relQ	0.31/1.01	0.63/1.08	0.71/1.01	0.79/1.01
Forgy 1 ×	0.8 ± 1.8	963.3 ± 6.5	975.3 ± 3.7	980.9 ± 4.6
1000 execs in s/relQ	0.3/1.01	0.63/1.08	0.71/1.01	0.79/1.01
MacQueen 1 ×	1.2 ± 3.2	961.8 ± 4.6	973.1 ± 5.2	982.9 ± 5.9
1000 execs in s/relQ	0.31/1.01	0.54/1.07	0.64/1.01	0.72/1
Hartigan–Wong 10 ×	0 ± 0	680.9 ± 25.2	1000 ± 0	1000 ± 0
1000 execs in s/relQ	0.81/1	7.23/1.05	10.24/1	13.42/1
Lloyd 10 ×	0 ± 0	678.6 ± 10.1	1000 ± 0	1000 ± 0
1000 execs in s/relQ	0.72/1	7.92/1.05	10.73/1	13.6/1
Forgy 10 ×	0 ± 0	666.7 ± 14.2	1000 ± 0	1000 ± 0
1000 execs in s/relQ	0.72/1	7.94/1.05	10.73/1	13.52/1
MacQueen 10 ×	0 ± 0	677.2 ± 12	1000 ± 0	1000 ± 0
1000 execs in s/relQ	0.73/1	7.15/1.05	9.99/1	12.92/1
Hartigan–Wong 20 ×	0 ± 0	457.5 ± 14	1000 ± 0	1000 ± 0
1000 execs in s/relQ	1.08/1	8.75/1.03	12.63/1	16.59/1
Lloyd 20 ×	0 ± 0	451.6 ± 9.5	1000 ± 0	1000 ± 0
1000 execs in s/relQ	0.9/1	9.96/1.03	13.58/1	16.79/1
Forgy 20 ×	0 ± 0	457.8 ± 12.4	1000 ± 0	1000 ± 0
1000 execs in s/relQ	0.9/1	9.95/1.03	13.44/1	16.78/1
MacQueen 20 ×	0 ± 0	452.6 ± 14.7	1000 ± 0	1000 ± 0
1000 execs in s/relQ	0.89/1	8.54/1.03	12.11/1	15.69/1
cmeans Fuzzy	0 ± 0	973.7 ± 7.9	995.4 ± 6.6	1000 ± 0
1000 execs in s/relQ	0.67/1	5.37/1.09	7.37/1.02	9.43/1.02
ufcl Fuzzy	235.5 ± 59.6	964.4 ± 8.9	976.8 ± 6.5	981.3 ± 5.4
1000 execs in s/relQ	0.8/1.52	6.3/1.14	8.79/1.06	11.22/1.05
single link	0 ± 0	0 ± 0	0 ± 0	0 ± 0
1000 execs in s/relQ	0.2/1	13.76/1	34.39/1.06	49.01/1.15
k -means++	0 ± 0	734.7 ± 17	811 ± 11.5	844 ± 12.5
1000 execs in s/relQ	1.14/1	16.01/1.05	22.53/1.01	28.77/1.02
k -means++ 2 ×	0 ± 0	544.7 ± 28.2	959.3 ± 6.1	976.9 ± 4.6
1000 execs in s/relQ	2.26/1	31.96/1.04	44.38/1	57.49/1

Other parameters fixed at $k = 2$ $d = 2$ $breakgap = 1$

between elements of P'_1 and P'_3 (and so respective gravity centres) are at least as big as between P_1 and P_3 , so that $SSC(C_A) - x_3^2 n_A b = SSC(P'_1 \cup P'_3) - x_3^2 n_A b \geq SSC(P'_1) + SSC(P'_3)$,

On the other hand, combining P'_1, P'_3 with disjoint parts P'_6, P'_7 of C_B will increase the sum of squared distances by at most $n_B x_5^2$, where x_5 is the distance between extreme elements of C_A and C_B : $SSC(P'_1 \cup P'_6) + SSC(P'_3 \cup P'_7) \leq SSC(P'_1) + |P_6| x_5^2 + SSC(P'_3) + |P_7| x_5^2 = SSC(P'_1) + SSC(P'_3) + n_B x_5^2$.

Combining these two relations, one gets

$$SSC(P'_1 \cup P'_6) + SSC(P'_3 \cup P'_7) \leq SSC(C_A) - x_3^2 n_A b + n_B x_5^2.$$

Therefore, as soon as one sets $b \geq \frac{n_B x_5^2}{(V_d/3/r)^2 n_A} \geq \frac{n_B x_5^2}{x_3^2 n_A}$, one will obtain

$$\begin{aligned} Q(\{P'_1 \cup P'_6, P'_3 \cup P'_7\}) \\ &= SSC(P'_1 \cup P'_6) + SSC(P'_3 \cup P'_7) \leq SSC(C_A) \\ &\leq SSC(C_A) + SSC(C_B) = Q(\{C_A, C_B\}) \end{aligned}$$

that is that for suitably large b it pays off to split C_A and merge C_B into parts of C_A , because the optimum lies at other partition than the one of well-separatedness in terms of big distance between centres of cluster enclosing balls. See also the discussion in “[Experimental Results](#)” section on Table 1.

Our Basic Approach to Clusterability

Let us stress at this point that the issue of well clusterability is not only a theoretical issue, but it is of practical interest too. For example when one intends to create synthetic data sets for investigating suitability of various clustering algorithms. But also after having performed the clustering process with whatever method one has. The important question needs to be answered: whether or not the obtained clustering meets the expectation of the analyst. These expectations may be divided into several categories:

- matching business goals,
- matching underlying algorithm assumptions,
- proximity to the optimal solutions.

Business goals of the clustering may be difficult to express in terms of data for an algorithm or may not fit the algorithm domain or data may be too expensive to collect prior to performing an approximate clustering. For example, when one seeks a clustering that would enable efficient collection of cars to be scrapped (disassembly network), then one has to match multiple goals, like covering the whole country, maximum distance from client to the disassembly station, and of course the number of prospective clients, which is known with some degree of uncertainty. The distances to the clients are frequently not Euclidean in nature (due to geographical obstacles like rivers, mountains, etc.), while the preferred k -means algorithm works best with geometrical distances, no upper distance can be imposed, etc. Other algorithms may induce same or different problems. So one has to check a posteriori if the obtained solution meets all criteria, does not violate constraints and is stable under fluctuation of the actual set of clients. The other two problems are somehow related to one another. For example, one may have clustered the data being a subsample of the proper data set and the question may be raised how close the subsample cluster centres are to the cluster centres of the proper data set. Known methods allow to estimate this discrepancy given that one knows that the cluster sizes do not differ too much. So prior to evaluating the correctness of cluster centre estimation one has to check if cluster proportions are within a required range (or if subsample size is relevant for such a verification). As another example consider methods of estimating closeness to optimal clustering solution under some general data distributions (like for the k -means++ [5]), but the guarantees are quite loose. But at the same time the guarantees can be much tighter if the clusters are well separated in some sense. So if one wants to be sure with a reasonable probability that the obtained solution is sufficiently close to the optimum, one would need to

check if the obtained clusters are well separated in the defined sense.

With this in mind, as mentioned, a number of researchers developed the concept of data clusterability. The notion of clusterability should intuitively reflect the following idea: if it is easy to see that there are clear-cut clusters in the data, then one would say that the data set is clusterable. “Easy to see” may mean either a visual inspection or some algorithm that quickly identifies the clusters. The well-established notion of clusterability would improve our understanding of the concept of the cluster itself—a well-defined clustering would be a clustering of clusterable points. This also would be a foundation for objective evaluation of clustering algorithms. The algorithm shall perform well for well-clusterable data and when the clusterability condition would be violated to some degree, the performance of a clustering algorithm is allowed to deteriorate also, but the algorithm quality would be measured on how the clusterability violation impacts the deterioration of algorithm performance.

However, the issue turns out not to be that simple. As is well known, each algorithm seeking to discover a clustering may be betrayed somehow to fail to discover a clustering structure that is visible upon human inspection of data. So instead of trying to reflect human vision of clusterability of the data set independently of the algorithm, let us rather concentrate on finding a concept of clusterability that is both reflecting human perception and the minimum of cost function of a concrete algorithm, in our case k -means. This paper concentrates particularly on its version called k -means++.

Let us define:

Definition 1 A data set is *well-clusterable* with respect to k -means cost function [Eq. (1)] for a given k (in brief: is *well-clusterable*) if (a) the data points may be split into subsets that are clearly separated by an appropriately chosen gap such that (b) the global minimum of k -means cost function coincides with this split and (c) with high probability (over 0.95) the k -means++ algorithm discovers this split and (d) if the split was found, it may be verified that the data subsets are separated by the abovementioned gap and (e) if the k -means++ did not discover a split of the data fulfilling the requirement of the existence of the gap, then with high probability the split described by points (a) and (b) does not exist.

Conditions, under which one can ensure that the minimum of k -means cost function is related to a clustering with (wide) gaps between clusters, were investigated in the paper [17].

The conditions for clusterable data set therein are rather rigid, but serve the purpose of demonstration that it is possible to define properties of the data set that ensure this

property of the minimum of k -means. Let us recall below the main result in this respect.

Assume that the data set encompassing n data points consists of k subsets such that each subset $i = 1, \dots, k$ can be enclosed in a ball of radius r_i . Let the gap (distance between surfaces of enclosing balls) between each pair of subsets amount to at least g , that is described below, as proposed in the paper [17] to ensure that the split of data into these subsets is the global optimum of k -means.

$$g \geq r_{\max} \sqrt{k \frac{M+n}{m}} \quad (2)$$

and

$$g \geq kr_{\max} \sqrt{n_p/2 + n_q/2 + n/2} \sqrt{\frac{2n}{n_p n_q}} \quad (3)$$

for any $p, q = 1, \dots, k; p \neq q$, when $n_i, i = 1, \dots, k$ is the cardinality of the cluster i , $M = \max_i n_i$, $m = \min_i n_i$. These conditions are taken from [17, formulas (21) and (22)], and they ensure that the global optimum agrees with a split into clusters separated by the mentioned gap.

Please note that the quotient of the cardinality of the largest to the smallest cluster increases the size of the required gap, as may be expected from “[Failure of Gaps as a Criterion for Clusterability](#)” section. It is visible from formula (2) that both the relationship M/m and n/m matter. This formula gives the impression that this relationship may be like square root of the sum of the two. But note that g is controlled also by formula (3), where the dependence of g on n/m may become close to linear, while that on M/m will still be close to square root. As visible from “[Failure of Gaps as a Criterion for Clusterability](#)” section, the sum of squared distances to cluster centre within the cluster and between clusters decides on the point when the shift in minimal costs occurs when the disproportion between cluster sizes grows. Hence g needs to grow as square root with this disproportion M/m . The impact of n/m shall be rather viewed in the context of the number of clusters k , as with fixed m and growing n n/m may be deemed as a reflection of k . If one looks at formula (5), one sees that g depends approximately quadratically on k . This relates probably to the fact that the number of possible misassignments between clusters grows quadratically with k .

It is claimed in [17] that the optimum of k -means objective is reached when splitting the data into the aforementioned subsets. The most fundamental implication is that the problem is decidable.

Theorem 1 (i) *If the data set is well-clusterable (into k clusters) with a gap defined by formulas (2) and (3), then with high probability k -means++ (after an appropriately*

chosen number of repetitions) will discover the respective clustering. (ii) If k -means++ (after an appropriately chosen number of repetitions) does not discover a clustering matching formulas (2) and (3), then with high probability the data set is not well clusterable with a gap defined by formulas (2) and (3).

The rest of the current section is devoted to the proof of the claims of this new theorem, proposed in the current paper.

If one obtained the split, then for each cluster one is able to compute the cluster centre, the radius of the ball containing all the data points of the cluster, and finally one can check if the gaps between the clusters meet the requirement of formulas (2) and (3). If this requirement is met, then definitely the data set is well clusterable.

Let us look at the claim (i). As shown in [17, section 12], the global minimum of k -means coincides with the separation by abovementioned gaps. Hence if there exists a positive probability, that k -means++ discovers the appropriate split, then the probability of finding the global minimum is increased by repeating independent runs of k -means++ and picking the split minimizing k -means cost function. It will be shown that the number of repetitions needed is known in advance, if the value of the quotient M/m is bounded from above by a known number.

First consider the easiest case of all clusters being of equal sizes ($M = m$). Then the above Eqs. (2) and (3) can be reduced to ($r = r_{\max}$)

$$g \geq r \sqrt{k(k+1)} \quad (4)$$

$$g \geq rk \sqrt{2k + k^2}. \quad (5)$$

A diagram of dependence of g/r on k is depicted in Fig. 2.

Now let us turn to k -means++ seeding. During initialization of the algorithm, the first cluster centre μ_1 is selected randomly (with uniform distribution) from among all the data points $\mathbf{X}$. The subsequent data points are selected randomly from the (remaining) data points $\mathbf{x} \in \mathbf{X}$ with probability $\text{Prob}(\mathbf{x}) = \frac{D(\mathbf{x}, \{\mu_1, \dots, \mu_i\})^2}{\sum_{\mathbf{y} \in \mathbf{X}} D(\mathbf{y}, \{\mu_1, \dots, \mu_i\})^2}$, where $D(\mathbf{x}, \{\mu_1, \dots, \mu_i\})$ denotes the distance of $\mathbf{x}$ to the closest point in the seed set $\{\mu_1, \dots, \mu_i\}$, if already i distinct clusters were seeded. Recall that with k clusters, we have km data points as all the clusters are of equal size. $(k-i)m$ of them are in clusters that do not have a seed yet, im are in those with a seed. The data points $\mathbf{x}$ in clusters without a seed have the distance to the closest seed of at least the size g . On the other hand, the data points $\mathbf{x}'$ in clusters with a seed have the distance to the closest seed of at most the size $2r$. Let $\mathbf{S} \subset \mathbf{X}$ be the set of data points in seeded clusters. Then the probability that a new cluster will be seeded

Fig. 2 Dependence of the relative gap g/r on k for clusters of equal radius and equal cardinalities

Fig. 3 Probability of seeding each cluster on k for clusters of equal radius and equal cardinalities

amounts to $\sum_{x \in X-S} \text{Prob}(X) = \sum_{x \in X-S} \frac{D(x, \{\mu_1, \dots, \mu_i\})^2}{\sum_{y \in X} D(y, \{\mu_1, \dots, \mu_i\})^2}$
 $= \frac{\sum_{x \in X-S} D(x, \{\mu_1, \dots, \mu_i\})^2}{\sum_{x \in X-S} D(x, \{\mu_1, \dots, \mu_i\})^2 + \sum_{y \in S} D(y, \{\mu_1, \dots, \mu_i\})^2} \geq \frac{\sum_{x \in X-S} D(x, \{\mu_1, \dots, \mu_i\})^2}{\sum_{x \in X-S} D(x, \{\mu_1, \dots, \mu_i\})^2 + im(2r)^2}$
 $\geq \frac{(k-i)mg^2}{(k-i)mg^2 + im(2r)^2} = \frac{(k-i)g^2}{(k-i)g^2 + 4ir^2}$. The last inequalities are justified first by substituting a positive number in the denominator by a bigger one, and then both in nominator and in denominator (that is bigger than the nominator) same number is substituted by the same smaller one.

Note that the inequality (5) implies $g \geq 2r\sqrt{k(k+1)}$ if $k \geq 2$ which we assume.

$$\begin{aligned} \frac{(k-i)g^2}{(k-i)g^2 + 4ir^2} &\geq \frac{(k-i)4r^2k^2(k+1)^2}{(k-i)4r^2k^2(k+1)^2 + i4r^2} \\ &= \frac{(k-i)k^2(k+1)^2}{(k-i)k^2(k+1)^2 + i} \geq \frac{k^2(k+1)^2}{k^2(k+1)^2 + (k-1)}. \end{aligned}$$

Hence the probability of accurate seeding ($\text{PAS}(k)$) amounts to

$$\text{PAS}(k) \geq \left(\frac{k^2(k+1)^2}{k^2(k+1)^2 + (k-1)} \right)^{k-1}$$

The diagram of dependence of this expression on k is depicted in Fig. 3.

Let us denote with Pr_{succ} the required probability of success in finding the global minimum. To ensure that the seeding was successful in Pr_{succ} (e.g. 95%) of cases,

k -means++ has to be rerun at least R times, with R given by

$$\begin{aligned} (1 - \text{PAS}(k))^R &\leq \left(1 - \left(\frac{k^2(k+1)^2}{k^2(k+1)^2 + (k-1)} \right)^{k-1} \right)^R \\ &< 1 - Pr_{\text{succ}} \\ R &\geq \frac{\log(1 - Pr_{\text{succ}})}{\log \left(1 - \left(\frac{k^2(k+1)^2}{k^2(k+1)^2 + (k-1)} \right)^{k-1} \right)} \end{aligned}$$

R does not depend on sample size But look at the following relationship:

$$\begin{aligned} &\left(\frac{k^2(k+1)^2}{k^2(k+1)^2 + (k-1)} \right)^{k-1} \\ &= \left(1 - \frac{k-1}{k^2(k+1)^2 + (k-1)} \right)^{k-1} \\ &= \left(1 - \frac{(k-1)^2}{k^2(k+1)^2 + (k-1)} \frac{1}{k-1} \right)^{k-1} \\ &\approx e^{-\frac{(k-1)^2}{k^2(k+1)^2 + (k-1)}}. \end{aligned}$$

The exponent of the last expression approaches rapidly zero, so that with increasing k within a single pass of k -means++ the optimum is reached. In fact, already for $k = 2$ an error of below 3% is obtained, for $k = 8$, below 1%, for $k = 30$ below 0.1%. See Fig. 3 for illustration.

Fig. 4 Dependence of the gap g for $k = 5$ for clusters of equal radius when varying cluster cardinalities

Let us discuss clusters with same radius, but different cardinalities. Let m be the cluster minimum cardinality, and M , respectively, the maximum.

$$g \geq r \sqrt{k \frac{M+n}{m}} \quad (6)$$

$$g \geq kr \sqrt{\frac{n(n_p + n_q + n)}{n_p n_q}} \quad (7)$$

for any $p, q = 1, \dots, k; p \neq q$, when $n_i, i = 1, \dots, k$ is the cardinality of the cluster i , $M = \max_i n_i$, $m = \min_i n_i$. Worst case g/r values are illustrated in Fig. 4.

Now let us turn to k -means++ seeding. Note that with $k \geq 4$ $g \geq 2r \sqrt{k \frac{M+n}{m}}$ from the above relations. If already i distinct clusters were seeded, then the probability that a new cluster will be seeded (under our assumptions) amounts to at least

$$\begin{aligned} & \frac{(k-i)mg^2}{(k-i)mg^2 + iM4r^2} \\ & \geq \frac{(k-i)mk^2n(1/m + 1/m + n/m^2)}{(k-i)mk^2n(1/m + 1/m + n/m^2) + iM} \\ & = \frac{(k-i)k^2n(2 + n/m)}{(k-i)k^2n(2 + n/m) + iM} \\ & \geq \frac{k^2n(2 + n/m)}{k^2n(2 + n/m) + (k-1)M}. \end{aligned}$$

Fig. 5 Dependence of the accurate seeding on the quotient M/m for $k = 5$

So again the probability of successful seeding will amount to at least:

$$\begin{aligned} & \left(\frac{k^2n(2 + n/m)}{k^2n(2 + n/m) + (k-1)M} \right)^{k-1} \\ & = \left(1 - \frac{(k-1)M}{k^2n(2 + n/m) + (k-1)M} \right)^{k-1} \\ & = \left(1 - \frac{(k-1)^2M}{k^2n(2 + n/m) + (k-1)M} \frac{1}{k-1} \right)^{k-1} \\ & \approx \exp \left(- \frac{(k-1)^2M}{k^2n(2 + n/m) + (k-1)M} \right). \end{aligned}$$

Even if M is 20 times as big as m , still the convergence to 1 is so rapid that already for $k = 2$ the clustering success is achieved with 95% success probability in a single repetition. An illustration is visible in Fig. 5.

It has been shown that if the data is well clusterable, then within practically a single clustering run the seeding will have the property that each cluster obtains a single seed. But what about the rest of the run of k -means? As in all these cases $g \geq 2r$, then, as shown in [17], the cluster centres will never switch to balls encompassing other clusters, so that eventually the true cluster structure is detected and minimum of Q is reached. This would complete the proof of claim (i). The demonstration of claim (ii) is straight forward. Note that if a clustering discovered by k -means fulfils the conditions of well-clusterability, then the data set is clusterable for sure, by definition. If the data were not well-clusterable, then k -means++ for

sure not find a clustering with the property of being well-clusterable, because it does not exist. If the data were well clusterable, then k -means++ would have failed to identify it with probability of at most $1 - \Pr_{\text{succ}}$.

So denote with W the event that the data is well-clusterable. Further denote with D the event that the k -means++ algorithm states that the data is well clusterable. It will be shown below that the probability $Pr(\neg W|\neg D)$ is high.

$$\begin{aligned} Pr(\neg W|\neg D) &= \frac{Pr(\neg D|\neg W)Pr(\neg W)}{Pr(\neg D|\neg W)Pr(\neg W) + Pr(\neg D|W)Pr(W)} \\ &= \frac{Pr(\neg W)}{Pr(\neg W) + Pr(\neg D|W)Pr(W)} \\ &= \frac{1}{1 + Pr(\neg D|W)\frac{Pr(W)}{Pr(\neg W)}} \\ &\geq \frac{1}{1 + (1 - \Pr_{\text{succ}})\frac{Pr(W)}{Pr(\neg W)}} \\ &\geq 1 - (1 - \Pr_{\text{succ}})\frac{Pr(W)}{Pr(\neg W)} \geq \Pr_{\text{succ}} \end{aligned}$$

The last inequality is true because the well-clusterable data are in practice extremely rare, and for sure less frequent than not well-clusterable ones.

Please note that a kind of worst-case analysis was presented above. It is obvious from this discussion that the probability of seeding of k distinct clusters depends on the characteristics of the data. The smallest gap between clusters was referred to in the above, but it may turn out that some clusters are stronger separated. This will automatically increase their probability of being hit so that the overall probability of hitting unhit clusters will increase significantly.

Smaller Gaps Between Clusters

Well-clusterability was considered in the previous section under the assumption of large areas between clusters where no data points of any cluster will occur. It will be shown subsequently that this assumption may be relaxed so that spurious points are allowed between the major concentrations of cluster points. But to ensure that the presence of such points will not lead the k -means procedure astray, core parts of the clusters will be distinguished and the subsequent Theorem 3 demonstrates that once a cluster core is hit by k -means initialization procedure, the cluster is preserved over subsequent k -means iterations.

It has been proven in [17, Theorem 17] that

Theorem 2 *Let A, B be cluster centres. Let ρ_{AB} be the radius of a ball centred at A and enclosing its cluster and it also*

is the radius of a ball centred at B and enclosing its cluster. If the distance between the cluster centres A, B amounts to $2\rho_{AB} + g$, $g > 0$ (g being the “gap” between clusters), if one picks any two points, X from the cluster of A and Y from the cluster of B , and recluster both clusters around X and Y , then the new clusters will preserve the balls centred at A and B of radius $g/2$ (called subsequently “cores”) each (X the core of A , Y the core of B).

The validity of a complementary theorem will be demonstrated here.

Theorem 3 *Let A, B be cluster centres. Let ρ_{AB} be the radius of a ball centred at A and enclosing its cluster and it also is the radius of a ball centred at B and enclosing its cluster. Let ρ_{cAB} be the radius of a ball centred at A and enclosing “vast majority” of its cluster and it also is the radius of a ball centred at B and enclosing “vast majority” of its cluster. If the distance between the cluster centres A, B amounts to $2\rho_{AB} + g$, $g > 0$ ($g = 2\rho_{cAB}$ being the “gap” between clusters), if one picks any two points, X from the ball $B(A, \rho_{cAB})$ and Y from the ball $B(B, \rho_{cAB})$, and recluster both clusters around X and Y , then the new clusters will be identical to the original clusters around A and B .*

Definition 2 If the gap between each pair of clusters fulfils the condition of either of the above two theorems, then we say that we have core-clustering.

Proof For the illustration of the proof see Fig. 6.

The proof does not differ too much from the previous one, and the previous Theorem 2 is a special case of Theorem 3.

Consider the two points A, B being the two centres of double balls. The inner ball represents the core of radius $\rho_{cAB} = g/2$, the outer ball of radius ρ ($\rho = \rho_{AB}$), enclosing the whole cluster. Consider two points, X, Y , one being in each core ball (presumably the cluster centres at some stage of the k -means algorithm). To represent their distances faithfully, a 3D space is needed at most.

Let us consider the plane established by the line AB and parallel to the line XY . Let X' and Y' be orthogonal projections of X, Y onto this plane. Now let us establish that the hyperplane π orthogonal to XY and passing through the middle of the line segment XY , that is the hyperplane containing the boundary between clusters centred at X and Y does not cut any of the balls centred at A and B . This hyperplane will be orthogonal to the plane of Fig. 6 and so it will manifest itself as an intersecting line l that should not cross outer circles around A and B , being projections of the respective balls. Let us draw two solid lines k, m between circles $O(A, \rho_{AB})$ and $O(B, \rho_{AB})$ tangential to each of them. Line l should lie between these lines, in which case the cluster centre will not jump to the other ball.

Fig. 6 An illustrative figure for the proof of the cluster preservation under a gap between cluster enclosing balls. The figure represents a projection of two clusters centred at A and B onto a plane containing the line AB . The inner circles centred at A and B contain the projections of cores of the respective clusters and outer circles contain the projections of whole clusters. The gap between clusters is represented by the circle with centre at O . It is proven that if the cluster centres change their positions (via the initialization procedure of k -means, for example) to any other place within the clusters A and B , then the core of A will remain in A , and the core of B will remain in B , in spite of the fact that other elements of the clusters may change cluster membership. Notation is explained in detail in the text

Let the line $X'Y'$ intersect with the circles $O(A, r_{cAB})$ and $O(B, r_{cAB})$ at points C, D, E, F as in the figure.

It is obvious that the line l would get closer to circle A , if the points X', Y' would lie closer to C and E , or closer to circle B if they would be closer to D and F .

Therefore, to show that it does not cut the circle $O(A, \rho)$ it is sufficient to consider $X' = C$ and $Y' = E$. (The case with ball $\text{Ball}(B, \rho)$ is symmetrical).

Let O be the centre of the line segment AB . Let us draw through this point a line parallel to CE that cuts the circles at points C', D', E' and F' . Now notice that centric symmetry through point O transforms the circles $O(A, r_{cAB}), O(B, r_{cAB})$ into one another, and point C' in F' and D' in E' . Let E^* and F^* be images of points E and F under this symmetry.

In order for the line l to lie between m and k , the middle point of the line segment CE shall lie between these lines.

Let us introduce a planar coordinate system centred at O with X axis parallel to lines m, k , such that A has both coordinates non-negative and B non-positive. Let us denote with α the angle between the lines AB and k . Given

the assumption that the distance between A and B equals $2\rho + 2r_{cAB}$, then the distance between lines k and m amounts to $2((\rho + r_{cAB}) \sin(\alpha) - \rho)$. Hence the Y coordinate of line k equals $((\rho + r_{cAB}) \sin(\alpha) - \rho)$.

So the Y coordinate of the centre of line segment CE shall be not higher than this. Let us express this in the coordinate system:

$$4(y_{OC} + y_{OE})/2 \leq ((\rho + r_{cAB}) \sin(\alpha) - \rho)$$

where y_{OC} is the y -coordinate of the vector $\overrightarrow{OC}$, etc.

Note, however, that

$$\begin{aligned} y_{OC} + y_{OE} &= y_{OA} + y_{AC} + y_{OB} + y_{BE} = y_{AC} + y_{BE} \\ &= y_{AC} - y_{AE^*} = y_{AC} + y_{E^*A}. \end{aligned}$$

So let us examine the circle with centre at A . Note that the lines CD and E^*F^* are at the same distance from the line $C'D'$. Note also that the absolute values of direction coefficients of tangentials of circle A at C' and D' are identical. The more distant these lines are, as line CD gets closer to A , the y_{AC} gets bigger, and y_{E^*A} becomes smaller. But from the properties of the circle, it is visible that y_{AC} increases at a decreasing rate, while y_{E^*A} decreases at an increasing rate. So the sum $y_{AC} + y_{E^*A}$ has the biggest value when C is identical with C' and it has to be proven only that

$$(y_{AC'} + y_{D'A})/2 = y_{AC'} \leq ((\rho + r_{cAB}) \sin(\alpha) - \rho).$$

Let M denote the middle point of the line segment $C'D'$. As point A has the coordinates $((\rho + r_{cAB}) \cos(\alpha), (\rho + r_{cAB}) \sin(\alpha))$, the point M is at distance of $(\rho + r_{cAB}) \cos(\alpha)$ from A . But $C'M^2 = r_{cAB}^2 - ((\rho + r_{cAB}) \cos(\alpha))^2$.

Hence, it is necessary to show that

$$r_{cAB}^2 - ((\rho + r_{cAB}) \cos(\alpha))^2 \leq ((\rho + r_{cAB}) \sin(\alpha) - \rho)^2$$

But this condition is equivalent to:

$$\begin{aligned} r_{cAB}^2 - ((\rho + r_{cAB}) \cos(\alpha))^2 &\leq ((\rho + r_{cAB}) \sin(\alpha))^2 \\ &+ \rho^2 - 2(\rho + r_{cAB})\rho \sin(\alpha) \\ r_{cAB}^2 &\leq (\rho + r_{cAB})^2 + \rho^2 - 2(\rho + r_{cAB})\rho \sin(\alpha) \\ r_{cAB}^2 - \rho^2 &\leq (\rho + r_{cAB})^2 - 2(\rho + r_{cAB})\rho \sin(\alpha) \\ (r_{cAB} - \rho)(r_{cAB} + \rho) &\leq (\rho + r_{cAB})^2 - 2(\rho + r_{cAB})(\rho) \sin(\alpha) \\ (r_{cAB} - \rho) &\leq (\rho + r_{cAB}) - 2\rho \sin(\alpha) \\ 0 &\leq 2\rho - 2\rho \sin(\alpha) \\ 0 &\leq 1 - \sin(\alpha) \end{aligned}$$

which is obviously true, as $\sin$ never exceeds 1. $\square$

Core-Based Global k -Means Minimum

In the paper [17], conditions have been investigated under which one can ensure that the minimum of k -means cost function is related to a clustering with (wide) gaps between clusters.

Based on the result of the preceding “**Smaller Gaps Between Clusters**” section, these conditions will be weakened requiring only that the big gaps exist between cluster cores and the clusters themselves are separated by much smaller gaps, equal to the size of the core.

In particular, let us consider the set of k clusters $\bar{C} = \{\bar{C}_1, \dots, \bar{C}_k\}$ of cardinalities $\bar{n}_1, \dots, \bar{n}_k$ and with radii of balls enclosing the clusters (with centres located at cluster centres) $\bar{r}_1, \dots, \bar{r}_k$. Let each of these clusters $\bar{C}_i$ have a core C_i around the cluster $\bar{C}_i$ centre of radius r_i and cardinality n_i such that for $\mathfrak{p} \in [0, 1)$

$$Q(\{C_i\})/Q(\{\bar{C}_i\}) \geq 1 - \mathfrak{p}.$$

One is interested in a gap g between cluster cores $C_1, \dots, C_k$ such that it does not make sense to split each cluster $\bar{C}_i$ into subclusters $\bar{C}_{i1}, \dots, \bar{C}_{ik}$ and to combine them into a set of new clusters $\mathcal{S} = \{S_1, \dots, S_k\}$ such that $S_j = \cup_{i=1}^k \bar{C}_{ij}$.

A g is to be found such that the highest possible central sum of squares combined over the clusters $\bar{C}_i$ would be lower than the lowest conceivable combined sums of squares around respective centres of clusters S_j . Let $\text{Var}(C)$ be the variance of the cluster C (average squared distance to set C gravity centre; with one exception, however: if referring to the *core* of any of the clusters $\bar{C}_i$, The cluster $\bar{C}_i$ gravity centre is computed, not the core C_i gravity centre, so also with the Q function). Let $C_{ij} = \bar{C}_{ij} \cap C_i$ be the core part of the subcluster $\bar{C}_{ij}$. Let r_{ij} be the distance of the centre of core subcluster C_{ij} to the centre of cluster $\bar{C}_i$. Let v_{ij} be the distance of the centre of core subcluster C_{ij} to the centre of core subcluster C_{ij} . So the total k -means function for the set of clusters $(C_1, \dots, C_k)$ will amount to:

$$Q(\bar{C}) = \frac{1}{1 - \mathfrak{p}} Q(C) = \frac{1}{1 - \mathfrak{p}} \sum_{i=1}^k \sum_{j=1}^k (n_{ij} \text{Var}(C_{ij}) + n_{ij} r_{ij}^2) \quad (8)$$

And the total k -means function for the set of clusters $(S_1, \dots, S_k)$ will amount to:

$$Q(\mathcal{S}) \geq \sum_{j=1}^k \left(\left(\sum_{i=1}^k n_{ij} \text{Var}(C_{ij}) \right) + \left(\sum_{i=1}^k n_{ij} \right) \left(\sum_{l=1}^{k-1} \sum_{i=l+1}^k \frac{n_{ij} n_{lj}}{\sum_{i=1}^k n_{ij} \sum_{l=1}^k n_{lj}} v_{ilj}^2 \right) \right). \quad (9)$$

Should $(\bar{C}_1, \dots, \bar{C}_k)$ constitute the absolute minimum of the k -means target function, then $Q(\mathcal{S}) \geq Q(\bar{C})$ should hold, which is fulfilled if:

$$\begin{aligned} & \sum_{j=1}^k \left(\left(\sum_{i=1}^k n_{ij} \text{Var}(C_{ij}) \right) + \left(\sum_{i=1}^k n_{ij} \right) \left(\sum_{l=1}^{k-1} \sum_{i=l+1}^k \frac{n_{ij} n_{lj}}{\sum_{i=1}^k n_{ij} \sum_{l=1}^k n_{lj}} v_{ilj}^2 \right) \right) \\ & \geq \frac{1}{1 - \mathfrak{p}} \sum_{i=1}^k \sum_{j=1}^k (n_{ij} \text{Var}(C_{ij}) + n_{ij} r_{ij}^2). \end{aligned}$$

Note that on the left-hand side of the inequality, the portion of the data outside of the cores has been ignored. This portion of the data would have made the left-hand side even bigger.

The above inequality is implied by:

$$\sum_{j=1}^k \left(\sum_{i=1}^{k-1} \sum_{l=i+1}^k \frac{n_{ij} n_{lj}}{\sum_{i=1}^k n_{ij}} v_{ilj}^2 \right) \geq \frac{1}{1 - \mathfrak{p}} \sum_{i=1}^k \sum_{j=1}^k (\mathfrak{p} n_{ij} \text{Var}(C_{ij}) + n_{ij} r_{ij}^2). \quad (10)$$

Note that $\text{Var}(C_{ij}) \leq r_{ij}^2$, so

$$\begin{aligned} \frac{1}{1 - \mathfrak{p}} \sum_{i=1}^k \sum_{j=1}^k (\mathfrak{p} n_{ij} \text{Var}(C_{ij}) + n_{ij} r_{ij}^2) & \leq \frac{1}{1 - \mathfrak{p}} \sum_{i=1}^k \sum_{j=1}^k (1 + \mathfrak{p}) n_{ij} r_{ij}^2 \\ & = \frac{1 + \mathfrak{p}}{1 - \mathfrak{p}} \sum_{i=1}^k \sum_{j=1}^k n_{ij} r_{ij}^2. \end{aligned} \quad (11)$$

To maximise $\sum_{j=1}^k n_{ij} r_{ij}^2$ for a single cluster C_i of enclosing ball radius r_i , note that you should set r_{ij} to r_i . Let $m_j = \arg \max_{j \in \{1, \dots, k\}} n_{ij}$. If one sets $r_{ij} = r_i$ for all j except m_j , then the maximal r_{im_j} is delimited by the relation $\sum_{j=1, j \neq m_j}^k n_{ij} r_{ij} \geq n_{im_j} r_{im_j}$. So

$$\begin{aligned} \sum_{j=1}^k n_{ij} r_{ij}^2 & \leq \left(\sum_{j=1, j \neq m_j}^k n_{ij} \right) r_i^2 \min \left(2, \left(1 + \frac{\sum_{j=1, j \neq m_j}^k n_{ij}}{n_{im_j}} \right) \right) \\ & \leq 2 \left(\sum_{j=1, j \neq m_j}^k n_{ij} \right) r_i^2. \end{aligned} \quad (12)$$

So if one can guarantee that the gap between cluster balls (of clusters from $\mathcal{C}$) amounts to g then surely

$$\sum_{j=1}^k \left(\sum_{i=1}^{k-1} \sum_{l=i+1}^k \frac{n_{ij} n_{lj}}{\sum_{i=1}^k n_{ij}} v_{ilj}^2 \right) \geq g^2 \sum_{j=1}^k \left(\sum_{i=1}^{k-1} \sum_{l=i+1}^k \frac{n_{ij} n_{lj}}{\sum_{i=1}^k n_{ij}} \right) \quad (13)$$

because in such case $g \leq v_{ilj}$ for all i, l, j .

By combining inequalities (10), (12) and (13), one sees that the global minimum is granted if the following holds:

$$g^2 \sum_{j=1}^k \left(\sum_{i=1}^{k-1} \sum_{l=i+1}^k \frac{n_{ij} n_{lj}}{\sum_{i=1}^k n_{ij}} \right) \geq 2 \frac{1 + \mathfrak{p}}{1 - \mathfrak{p}} \sum_{i=1}^k \left(\sum_{j=1, j \neq m_j}^k n_{ij} \right) r_i^2. \quad (14)$$

One can distinguish two cases: either (1) there exists a cluster S_i containing two subclusters C_{pi}, C_{qt} such that

$t = \arg \max_j |C_{pj}|$ and $t = \arg \max_j |C_{qj}|$ (maximum cardinality subclasses of their respective original clusters C_p, C_q or (2) not.

Consider the first case. Let C_p, C_q be the two clusters where C_{pt} and C_{qt} be two subclusters of highest cardinality within C_p, C_q resp. This implies that $n_{pt} \geq \frac{1}{k} n_p, n_{qt} \geq \frac{1}{k} n_q$. Also this implies that for $i \neq p, i \neq q, n_{it} \leq n_i/2$.

$$\begin{aligned} \sum_{j=1}^k \sum_{i=1}^{k-1} \sum_{l=i+1}^k \frac{n_{ij} n_{lj}}{\sum_{i=1}^k n_{ij}} &\geq \sum_{i=1}^{k-1} \sum_{l=i+1}^k \frac{n_{it} n_{lt}}{\sum_{i=1}^k n_{it}} \geq \frac{n_{pt} n_{qt}}{\sum_{i=1}^k n_{it}} \\ &\geq \frac{n_{pt} n_{qt}}{n_p/2 + n_q/2 + \sum_{i=1}^k n_i/2} = \frac{n_{pt} n_{qt}}{n_p/2 + n_q/2 + n/2} \\ &\geq \frac{1}{k^2} \frac{n_p n_q}{n_p/2 + n_q/2 + n/2}. \end{aligned}$$

Note that

$$2 \sum_{i=1}^k \left(\sum_{j=1, j \neq m_j}^k n_{ij} \right) r_i^2 \leq 2 \sum_{i=1}^k n_i r_i^2$$

So, in order to fulfil inequality (14), it is sufficient to require that

$$\begin{aligned} g &\geq \sqrt{\frac{2 \frac{1+p}{1-p} \sum_{i=1}^k n_i r_i^2}{\frac{1}{k^2} \frac{n_p n_q}{n_p/2 + n_q/2 + n/2}}} \\ &= k \sqrt{n_p/2 + n_q/2 + n/2} \sqrt{\frac{2 \frac{1+p}{1-p} \sum_{i=1}^k n_i r_i^2}{n_p n_q}} \quad (15) \\ &= k \sqrt{n_p + n_q + n} \sqrt{\frac{\frac{1+p}{1-p} \sum_{i=1}^k n_i r_i^2}{n_p n_q}} \end{aligned}$$

This of course maximized over all combinations of p, q .

Let us proceed to the second case. Here each cluster S_j contains a subcluster of maximum cardinality of a different cluster C_i . As the relation between S_j and C_i is unique, one can reindex S_j in such a way that actually C_j contains its maximum cardinality subcluster C_{jj} . Let us rewrite the inequality (14).

$$g^2 \sum_{j=1}^k \left(\sum_{i=1}^{k-1} \sum_{l=i+1}^k \frac{n_{ij} n_{lj}}{\sum_{i=1}^k n_{ij}} \right) - 2 \frac{1+p}{1-p} \sum_{i=1}^k \left(\sum_{j=1, j \neq m_j}^k n_{ij} \right) r_i^2 \geq 0.$$

This is met if

$$g^2 \sum_{j=1}^k \left(\sum_{i=1}^{j-1} \frac{n_{ij} n_{jj}}{\sum_{i=1}^k n_{ij}} + \sum_{l=j+1}^k \frac{n_{lj} n_{jj}}{\sum_{l=1}^k n_{lj}} \right) - 2 \frac{1+p}{1-p} \sum_{i=1}^k (n_i - n_{ii}) r_i^2 \geq 0$$

This is the same as:

$$g^2 \sum_{j=1}^k \left(\sum_{i=1, \dots, j-1, j+1, \dots, k} \frac{n_{ij} n_{jj}}{\sum_{i=1}^k n_{ij}} \right) - 2 \frac{1+p}{1-p} \sum_{i=1}^k (n_i - n_{ii}) r_i^2 \geq 0.$$

This is fulfilled if:

$$\begin{aligned} g^2 \sum_{j=1}^k \left(\sum_{i=1, \dots, j-1, j+1, \dots, k} \frac{n_{ij} n_{jj}/k}{n_j/2 + \sum_{i=1}^k n_i/2} \right) \\ - 2 \frac{1+p}{1-p} \sum_{i=1}^k (n_i - n_{ii}) r_i^2 \geq 0. \end{aligned}$$

Let M be the maximum over $n_1, \dots, n_k$. The above holds if

$$g^2 \sum_{j=1}^k \left(\sum_{i=1, \dots, j-1, j+1, \dots, k} \frac{n_{ij} n_{jj}/k}{M/2 + n/2} \right) - 2 \frac{1+p}{1-p} \sum_{i=1}^k (n_i - n_{ii}) r_i^2 \geq 0$$

Let m be the minimum over $n_1, \dots, n_k$. The above holds if

$$g^2 \sum_{j=1}^k \left(\sum_{i=1, \dots, j-1, j+1, \dots, k} \frac{n_{ij} m/k}{M/2 + n/2} \right) - 2 \frac{1+p}{1-p} \sum_{i=1}^k (n_i - n_{ii}) r_i^2 \geq 0$$

This is the same as

$$\begin{aligned} g^2 \frac{m/k}{M/2 + n/2} \left(\sum_{j=1}^k \sum_{i=1, \dots, j-1, j+1, \dots, k} n_{ij} \right) \\ - 2 \frac{1+p}{1-p} \sum_{i=1}^k (n_i - n_{ii}) r_i^2 \geq 0 \\ g^2 \frac{m/k}{M/2 + n/2} \left(\sum_{j=1}^k \left(\left(\sum_{i=1}^k n_{ij} \right) - n_{jj} \right) \right) \\ - 2 \frac{1+p}{1-p} \sum_{i=1}^k (n_i - n_{ii}) r_i^2 \geq 0 \end{aligned}$$

$$g^2 \frac{m/k}{M/2 + n/2} \left(\left(\sum_{j=1}^k \sum_{i=1}^k n_{ij} \right) - \left(\sum_{j=1}^k n_{jj} \right) \right) \\ - 2 \frac{1+p}{1-p} \sum_{i=1}^k (n_i - n_{ii}) r_i^2 \geq 0$$

$$- 2 \frac{1+p}{1-p} \sum_{i=1}^k (n_i - n_{ii}) r_i^2 \geq 0$$

$$g^2 \frac{m/k}{M/2 + n/2} \left(\left(\sum_{i=1}^k n_i \right) - \left(\sum_{j=1}^k n_{jj} \right) \right) \\ - 2 \frac{1+p}{1-p} \sum_{i=1}^k (n_i - n_{ii}) r_i^2 \geq 0$$

$$- 2 \frac{1+p}{1-p} \sum_{i=1}^k (n_i - n_{ii}) r_i^2 \geq 0$$

$$g^2 \frac{m/k}{M/2 + n/2} \left(\sum_{i=1}^k (n_i - n_{ii}) \right) \\ - 2 \frac{1+p}{1-p} \sum_{i=1}^k (n_i - n_{ii}) r_i^2 \geq 0$$

$$- 2 \frac{1+p}{1-p} \sum_{i=1}^k (n_i - n_{ii}) r_i^2 \geq 0$$

$$\sum_{i=1}^k (n_i - n_{ii}) \left(g^2 \frac{m/k}{M/2 + n/2} - 2 \frac{1+p}{1-p} r_i^2 \right) \geq 0$$

The above will hold, if for every $i = 1, \dots, k$

$$g \geq r_i \sqrt{\frac{1+p}{1-p} \frac{2}{\frac{m/k}{M/2+n/2}}} \quad (16)$$

$$g \geq r_i \sqrt{k \frac{1+p}{1-p} \frac{M+n}{m}}.$$

So the inequality (14) is fulfilled, if both inequality (15) and inequality (16) are held by an appropriately chosen g .

In summary, it has been shown that

Theorem 4 Let $\bar{C} = \{\bar{C}_1, \dots, \bar{C}_k\}$ be a partition of a data set into k clusters of cardinalities $\bar{n}_1, \dots, \bar{n}_k$ and with radii of balls enclosing the clusters (with centres located at cluster centres) $\bar{r}_1, \dots, \bar{r}_k$. Let each of these clusters $\bar{C}_i$ have a core C_i of radius r_i and cardinality n_i around the cluster centre such that for $p \in [0, 1)$

$$Q(\{C_i\})/Q(\{\bar{C}_i\}) \geq 1-p$$

Then if the gap g between cluster cores $C_1, \dots, C_k$ fulfils conditions expressed in formulas (15) and (16) then the partition $\bar{C}$ coincides with the global minimum of the k -means cost function for the data set.

Core-Based Approach to Clusterability

After the preceding preparatory work, a theorem analogous to Theorem 1 will be proven, but now allowing for smaller gaps between clusters.

Theorem 5 (i) If the data set is well-clusterable with a gap defined by formulas (16) and (15), with r_i replaced by their maxima, then with high probability k -means++ (after an appropriately chosen number of repetitions) will discover the respective clustering. (ii) If k -means++ (after an appropriately chosen number of repetitions) does not discover a clustering matching formulas (16) and (15) (with r_i replaced by their maxima), then with high probability the data set is not well clusterable with a gap defined by formulas (16) and (15).

The rest of the current section is devoted to the proof of the claims of this theorem.

If one obtained the split, then one is able to compute for each cluster the cluster centre, the radius of the ball containing all the data points of the cluster but the most distant ones, constituting at most p of the quality function for the cluster, and finally one can check if the gaps between the

cluster cores meet the requirement of formulas (16) and (15). So one is able to decide that one has found that the data set is well clusterable.

So let us look at the claim (i). As demonstrated in preceding “Core-Based Global k -Means Minimum” section, the global minimum of k -means coincides with the separation by abovementioned gaps. Hence if there exists a positive probability, that k -means++ discovers the appropriate split, then by repeating independent runs of k -means++ and picking the split minimizing k -means cost function one will increase the probability of finding the global minimum. It will be shown that one can know the number of repetitions needed in advance, if an upper bound of the quotient M/m is assumed.

Let us assume it is granted that

$$g \geq r \sqrt{k \frac{1+p}{1-p} \frac{M+n}{m}} \quad (17)$$

for any $i = 1, \dots, k$

$$g \geq kr \sqrt{\frac{1+p}{1-p} \frac{n(n_p + n_q + n)}{n_p n_q}} \quad (18)$$

for any $p, q = 1, \dots, k; p \neq q$, when $n_i, i = 1, \dots, k$ is the cardinality of the cluster i , $M = \max_i n_i$, $m = \min_i n_i$. For an illustration of this dependence see Fig. 7.

So let us turn to k -means++ seeding. If already i distinct cluster cores were seeded, then the probability that a new cluster core will be seeded (under our assumptions, when $k \geq 4$ as previously) amounts to at least

Fig. 7 Dependence of g/r for $k = 5$ on the value of p

Fig. 8 Dependence of the accurate seeding on share p of variance outside the core for $k = 5$ and $M/m = 4$

$$\begin{aligned}
 & \frac{(k-i)m(1-p)g^2}{(k-i)mg^2 + iM\frac{1}{1-p}4r^2} \\
 & \geq \frac{(k-i)mk^2(1-p)\frac{1+p}{1-p}n(1/m + 1/m + n/m^2)}{(k-i)mk^2\frac{1+p}{1-p}n(1/m + 1/m + n/m^2) + iM\frac{1}{1-p}} \\
 & = \frac{(k-i)k^2(1-p)(1+p)n(2+n/m)}{(k-i)k^2(1+p)n(2+n/m) + iM} \\
 & \geq \frac{k^2(1-p)(1+p)n(2+n/m)}{k^2(1+p)n(2+n/m) + (k-1)M}
 \end{aligned}$$

So again the probability of successful seeding will amount to at least:

$$\begin{aligned}
 & \left(\frac{k^2(1-p)(1+p)n(2+n/m)}{k^2(1+p)n(2+n/m) + (k-1)M} \right)^{k-1} \\
 & = (1-p)^{k-1} \left(1 - \frac{(k-1)M}{k^2(1+p)n(2+n/m) + (k-1)M} \right)^{k-1} \\
 & = (1-p)^{k-1} \left(1 - \frac{(k-1)^2M}{k^2(1+p)n(2+n/m) + (k-1)M} \frac{1}{k-1} \right)^{k-1} \\
 & \approx (1-p)^{k-1} \exp \left(- \frac{(k-1)^2M}{k^2(1+p)n(2+n/m) + (k-1)M} \right).
 \end{aligned}$$

For an illustration of this dependence see Fig. 8.

Apparently in the limit the above expression lies at about $(1-p)^{k-1}$.

Fig. 9 Repetitions needed to get accurate seeding on share p of variance outside the core for $k = 5$ and $M/m = 4$

So to achieve the identification of the clustering with probability of at least $\Pr_{\text{succ}}$ (e.g. 95%), one will need R runs of k -means++ where

$$R = \frac{\log(1 - \Pr_{\text{succ}})}{\log(1 - (1-p)^{k-1})}.$$

Note that

$$1 - (1-p)^{k-1} \approx 1 - e^{-p(k-1)} \approx 1 - e^{-pk}.$$

The effect of doubling k is

$$\frac{1 - e^{-p2k}}{1 - e^{-pk}} = \frac{(1 - e^{-pk})(1 + e^{-p2k})}{1 - e^{-pk}} = 1 + e^{-p2k}$$

that is it is sublinear in the expression $1 - (1-p)^{k-1}$, hence R grows slower than reciprocally logarithmically in k and p . For an illustration of this relation, see Fig. 9.

So far we have concentrated on showing that if the data is well clusterable, then within practically reasonable number of k -means++ runs the seeding will have the property that each cluster obtains a single seed. But what about the rest of the run of k -means? As shown in “[Smaller Gaps Between Clusters](#)” section, the cluster centres will never switch to balls encompassing other clusters, so that eventually the true cluster structure is detected and minimum of Q is reached. This would complete the proof of claim (i). The demonstration of claim (ii) is straight forward. If the data were well clusterable then k -means++ would have failed to identify it with probability of at most $1 - \Pr_{\text{succ}}$. As the well-clusterable data are in practice extremely rare, the failure of the algorithm to identify a well-clusterable structure induces with probability of at least $\Pr_{\text{succ}}$

Table 2 Reconstruction of the clusters in synthetic data. Notation in text

Dataset	Errors		WC disc.		WC not disc.	
	<i>k</i> -means	<i>k</i> -means++	cc	wc	cc	wc
$k = 2, p = 0, gp = 0.5$	0	0	0	0	100	0
$k = 2, p = 0.1, gp = 0.5$	0	0	0	0	100	0
$k = 3, p = 0, gp = 0.5$	38	0	0	0	100	0
$k = 3, p = 0.1, gp = 0.5$	26	0	0	0	100	0
$k = 5, p = 0, gp = 0.5$	74	0	0	0	100	0
$k = 5, p = 0.1, gp = 0.5$	66	0	0	0	100	0
$k = 2, p = 0, gp = 0.7$	0	0	0	0	100	0
$k = 2, p = 0.1, gp = 0.7$	0	0	0	0	100	0
$k = 3, p = 0, gp = 0.7$	50	2	0	0	98	2
$k = 3, p = 0.1, gp = 0.7$	37	0	0	0	100	0
$k = 5, p = 0, gp = 0.7$	78	0	0	0	100	0
$k = 5, p = 0.1, gp = 0.7$	73	0	0	0	100	0
$k = 2, p = 0, gp = 0.9$	0	0	0	0	100	0
$k = 2, p = 0.1, gp = 0.9$	0	0	0	0	100	0
$k = 3, p = 0, gp = 0.9$	41	1	0	0	99	1
$k = 3, p = 0.1, gp = 0.9$	29	1	0	0	99	1
$k = 5, p = 0, gp = 0.9$	74	0	0	0	100	0
$k = 5, p = 0.1, gp = 0.9$	78	0	0	0	100	0
$k = 2, p = 0, gp = 1.1$	0	0	100	0	0	0
$k = 2, p = 0.1, gp = 1.1$	0	0	100	0	0	0
$k = 3, p = 0, gp = 1.1$	41	0	100	0	0	0
$k = 3, p = 0.1, gp = 1.1$	39	0	100	0	0	0
$k = 5, p = 0, gp = 1.1$	80	0	100	0	0	0
$k = 5, p = 0.1, gp = 1.1$	80	0	100	0	0	0
$k = 2, p = 0, gp = 1.3$	0	0	100	0	0	0
$k = 2, p = 0.1, gp = 1.3$	0	0	100	0	0	0
$k = 3, p = 0, gp = 1.3$	43	0	100	0	0	0
$k = 3, p = 0.1, gp = 1.3$	38	0	100	0	0	0
$k = 5, p = 0, gp = 1.3$	80	0	100	0	0	0
$k = 5, p = 0.1, gp = 1.3$	75	0	100	0	0	0
$k = 2, p = 0, gp = 1.5$	0	0	100	0	0	0
$k = 2, p = 0.1, gp = 1.5$	0	0	100	0	0	0
$k = 3, p = 0, gp = 1.5$	33	1	99	0	0	1
$k = 3, p = 0.1, gp = 1.5$	33	0	100	0	0	0
$k = 5, p = 0, gp = 1.5$	75	0	100	0	0	0
$k = 5, p = 0.1, gp = 1.5$	79	0	100	0	0	0

that no such structure exists in the data. A detailed proof follows the reasoning of the last part of the proof of Theorem 1.

Experimental Results

Goals of Experiments

In order to illustrate the issues raised in this paper, three types of experiments were performed. The first experiment, performed on synthetic data, is devoted to the mismatch of gaps between subsets of data and the clusterings obtained

by common clustering algorithms. The results are shown in Table 1. The second, performed on synthetic data (Table 2), and third, performed on real data (Table 3), are devoted to demonstration that *k*-means++ is able to discover well-clusterable data. In particular, it is shown that: (1) If a dataset is well clusterable as defined in Theorems 1 or 5 (based on Definition 1) then *k*-means++ is able to identify the best clustering (both for real-world datasets and synthetic ones), (2) If *k*-means++ cannot find a clustering satisfying well-clusterability, there is no good clustering structure, fitting those definitions, hidden in data (with high probability) for all *k*-means style algorithms.

Table 3 Reconstruction of the clusters in real data from *R* library *datasets*

Dataset	Gap type	Clusters	p	Errors		WC disc.		WC not disc.	
				<i>k</i> -means	<i>k</i> -means++	cc	wc	cc	wc
DNase	orig	2	0	69	39	0	0	61	39
DNase	orig	2	0.1	59	38	0	0	62	38
DNase	<i>g</i> /2	2	0	0	0	0	0	100	0
DNase	<i>g</i> /2	2	0.1	0	0	0	0	100	0
DNase	<i>g</i>	2	0	0	0	100	0	0	0
DNase	<i>g</i>	2	0.1	0	0	100	0	0	0
DNase	2 <i>g</i>	2	0	0	0	100	0	0	0
DNase	2 <i>g</i>	2	0.1	0	0	100	0	0	0
DNase	orig	3	0	81	25	0	0	75	25
DNase	orig	3	0.1	80	28	0	0	72	28
DNase	<i>g</i> /2	3	0	63	1	0	0	99	1
DNase	<i>g</i> /2	3	0.1	59	1	0	0	99	1
DNase	<i>g</i>	3	0	58	0	100	0	0	0
DNase	<i>g</i>	3	0.1	51	0	100	0	0	0
DNase	2 <i>g</i>	3	0	50	0	100	0	0	0
DNase	2 <i>g</i>	3	0.1	49	0	100	0	0	0
DNase	orig	5	0	75	35	0	0	65	35
DNase	orig	5	0.1	79	30	0	0	70	30
DNase	<i>g</i> /2	5	0	84	0	0	0	100	0
DNase	<i>g</i> /2	5	0.1	86	0	0	0	100	0
DNase	<i>g</i>	5	0	80	0	100	0	0	0
DNase	<i>g</i>	5	0.1	79	0	100	0	0	0
DNase	2 <i>g</i>	5	0	83	0	100	0	0	0
DNase	2 <i>g</i>	5	0.1	81	0	100	0	0	0
iris	orig	2	0	0	0	0	0	100	0
iris	orig	2	0.1	0	0	0	0	100	0
iris	<i>g</i> /2	2	0	0	0	0	0	100	0
iris	<i>g</i> /2	2	0.1	0	0	0	0	100	0
iris	<i>g</i>	2	0	0	0	100	0	0	0
iris	<i>g</i>	2	0.1	0	0	100	0	0	0
iris	2 <i>g</i>	2	0	0	0	100	0	0	0
iris	2 <i>g</i>	2	0.1	0	0	100	0	0	0
iris	orig	3	0	19	17	0	0	83	17
iris	orig	3	0.1	23	15	0	0	85	15
iris	<i>g</i> /2	3	0	35	0	0	0	100	0
iris	<i>g</i> /2	3	0.1	40	0	0	0	100	0
iris	<i>g</i>	3	0	40	0	100	0	0	0
iris	<i>g</i>	3	0.1	24	0	100	0	0	0
iris	2 <i>g</i>	3	0	36	0	100	0	0	0
iris	2 <i>g</i>	3	0.1	31	0	100	0	0	0
iris	orig	5	0	94	97	0	0	3	97
iris	orig	5	0.1	96	95	0	0	5	95
iris	<i>g</i> /2	5	0	69	0	0	0	100	0
iris	<i>g</i> /2	5	0.1	72	0	0	0	100	0
iris	<i>g</i>	5	0	71	0	0	0	100	0
iris	<i>g</i>	5	0.1	68	0	0	0	100	0
iris	2 <i>g</i>	5	0	72	0	100	0	0	0
iris	2 <i>g</i>	5	0.1	71	0	100	0	0	0
Theoph	orig	2	0	8	29	0	0	71	29

Table 3 (continued)

Dataset	Gap type	Clusters	p	Errors		WC disc.		WC not disc.	
				k-means	k-means++	cc	wc	cc	wc
Theoph	orig	2	0.1	20	17	0	0	83	17
Theoph	g/2	2	0	0	0	0	0	100	0
Theoph	g/2	2	0.1	0	0	0	0	100	0
Theoph	g	2	0	0	0	100	0	0	0
Theoph	g	2	0.1	0	0	100	0	0	0
Theoph	2g	2	0	0	0	100	0	0	0
Theoph	2g	2	0.1	0	0	100	0	0	0
Theoph	orig	3	0	83	81	0	0	19	81
Theoph	orig	3	0.1	83	78	0	0	22	78
Theoph	g/2	3	0	21	0	0	0	100	0
Theoph	g/2	3	0.1	32	0	0	0	100	0
Theoph	g	3	0	32	0	100	0	0	0
Theoph	g	3	0.1	35	0	100	0	0	0
Theoph	2g	3	0	34	1	99	0	0	1
Theoph	2g	3	0.1	28	0	100	0	0	0
Theoph	orig	5	0	84	84	0	0	16	84
Theoph	orig	5	0.1	84	80	0	0	20	80
Theoph	g/2	5	0	68	0	0	0	100	0
Theoph	g/2	5	0.1	73	0	0	0	100	0
Theoph	g	5	0	72	0	100	0	0	0
Theoph	g	5	0.1	72	0	100	0	0	0
Theoph	2g	5	0	72	0	100	0	0	0
Theoph	2g	5	0.1	74	0	100	0	0	0
attenu	orig	2	0	44	42	0	0	58	42
attenu	orig	2	0.1	41	49	0	0	51	49
attenu	g/2	2	0	0	0	0	0	100	0
attenu	g/2	2	0.1	0	0	0	0	100	0
attenu	g	2	0	0	0	100	0	0	0
attenu	g	2	0.1	0	0	100	0	0	0
attenu	2g	2	0	0	0	100	0	0	0
attenu	2g	2	0.1	0	0	100	0	0	0
attenu	orig	3	0	50	61	0	0	39	61
attenu	orig	3	0.1	50	58	0	0	42	58
attenu	g/2	3	0	33	0	0	0	100	0
attenu	g/2	3	0.1	38	2	0	0	98	2
attenu	g	3	0	43	0	0	0	100	0
attenu	g	3	0.1	39	0	0	0	100	0
attenu	2g	3	0	41	0	100	0	0	0
attenu	2g	3	0.1	36	0	100	0	0	0
attenu	orig	5	0	80	76	0	0	24	76
attenu	orig	5	0.1	79	70	0	0	30	70
attenu	g/2	5	0	84	0	0	0	100	0
attenu	g/2	5	0.1	89	0	0	0	100	0
attenu	g	5	0	75	0	0	0	100	0
attenu	g	5	0.1	85	0	0	0	100	0
attenu	2g	5	0	82	0	100	0	0	0
attenu	2g	5	0.1	78	0	100	0	0	0
faithful	orig	2	0	0	0	0	0	100	0
faithful	orig	2	0.1	0	0	0	0	100	0

Table 3 (continued)

Dataset	Gap type	Clusters	p	Errors		WC disc.		WC not disc.	
				<i>k</i> -means	<i>k</i> -means++	cc	wc	cc	wc
faithful	g/2	2	0	0	0	0	0	100	0
faithful	g/2	2	0.1	0	0	0	0	100	0
faithful	g	2	0	0	0	100	0	0	0
faithful	g	2	0.1	0	0	100	0	0	0
faithful	2g	2	0	0	0	100	0	0	0
faithful	2g	2	0.1	0	0	100	0	0	0
faithful	orig	3	0	59	60	0	0	40	60
faithful	orig	3	0.1	59	62	0	0	38	62
faithful	g/2	3	0	25	0	0	0	100	0
faithful	g/2	3	0.1	29	0	0	0	100	0
faithful	g	3	0	27	0	100	0	0	0
faithful	g	3	0.1	28	0	100	0	0	0
faithful	2g	3	0	29	0	100	0	0	0
faithful	2g	3	0.1	25	0	100	0	0	0
faithful	orig	5	0	72	47	0	0	53	47
faithful	orig	5	0.1	65	57	0	0	43	57
faithful	g/2	5	0	72	0	0	0	100	0
faithful	g/2	5	0.1	77	0	0	0	100	0
faithful	g	5	0	81	0	100	0	0	0
faithful	g	5	0.1	76	0	100	0	0	0
faithful	2g	5	0	77	0	100	0	0	0
faithful	2g	5	0.1	74	0	100	0	0	0
infert	orig	2	0	95	94	0	0	6	94
infert	orig	2	0.1	96	94	0	0	6	94
infert	g/2	2	0	0	0	0	0	100	0
infert	g/2	2	0.1	0	0	0	0	100	0
infert	g	2	0	0	0	100	0	0	0
infert	g	2	0.1	0	0	100	0	0	0
infert	2g	2	0	0	0	100	0	0	0
infert	2g	2	0.1	0	0	100	0	0	0
infert	orig	3	0	90	92	0	0	8	92
infert	orig	3	0.1	95	88	0	0	12	88
infert	g/2	3	0	34	1	0	0	99	1
infert	g/2	3	0.1	31	0	0	0	100	0
infert	g	3	0	27	0	0	0	100	0
infert	g	3	0.1	35	0	0	0	100	0
infert	2g	3	0	38	1	99	0	0	1
infert	2g	3	0.1	26	0	100	0	0	0
infert	orig	5	0	82	83	0	0	17	83
infert	orig	5	0.1	81	92	0	0	8	92
infert	g/2	5	0	75	1	0	0	99	1
infert	g/2	5	0.1	76	0	0	0	100	0
infert	g	5	0	86	0	0	0	100	0
infert	g	5	0.1	79	0	0	0	100	0
infert	2g	5	0	79	0	100	0	0	0
infert	2g	5	0.1	71	0	100	0	0	0
quakes	orig	2	0	31	34	0	0	66	34
quakes	orig	2	0.1	29	23	0	0	77	23
quakes	g/2	2	0	0	0	0	0	100	0

Table 3 (continued)

Dataset	Gap type	Clusters	p	Errors		WC disc.		WC not disc.	
				<i>k</i> -means	<i>k</i> -means++	cc	wc	cc	wc
quakes	g/2	2	0.1	0	0	0	0	100	0
quakes	g	2	0	0	0	100	0	0	0
quakes	g	2	0.1	0	0	100	0	0	0
quakes	2g	2	0	0	0	100	0	0	0
quakes	2g	2	0.1	0	0	100	0	0	0
quakes	orig	3	0	0	0	0	0	100	0
quakes	orig	3	0.1	0	0	0	0	100	0
quakes	g/2	3	0	31	0	0	0	100	0
quakes	g/2	3	0.1	35	0	0	0	100	0
quakes	g	3	0	32	0	0	0	100	0
quakes	g	3	0.1	25	0	0	0	100	0
quakes	2g	3	0	33	0	100	0	0	0
quakes	2g	3	0.1	34	0	100	0	0	0
quakes	orig	5	0	93	85	0	0	15	85
quakes	orig	5	0.1	81	88	0	0	12	88
quakes	g/2	5	0	78	2	0	0	98	2
quakes	g/2	5	0.1	70	0	0	0	100	0
quakes	g	5	0	78	0	0	0	100	0
quakes	g	5	0.1	80	0	0	0	100	0
quakes	2g	5	0	80	0	0	0	100	0
quakes	2g	5	0.1	77	0	100	0	0	0
randu	orig	2	0	62	69	0	0	31	69
randu	orig	2	0.1	63	64	0	0	36	64
randu	g/2	2	0	0	0	0	0	100	0
randu	g/2	2	0.1	0	0	0	0	100	0
randu	g	2	0	0	0	100	0	0	0
randu	g	2	0.1	0	0	100	0	0	0
randu	2g	2	0	0	0	100	0	0	0
randu	2g	2	0.1	0	0	100	0	0	0
randu	orig	3	0	93	95	0	0	5	95
randu	orig	3	0.1	97	93	0	0	7	93
randu	g/2	3	0	16	0	0	0	100	0
randu	g/2	3	0.1	21	0	0	0	100	0
randu	g	3	0	34	0	0	0	100	0
randu	g	3	0.1	29	0	0	0	100	0
randu	2g	3	0	30	0	100	0	0	0
randu	2g	3	0.1	41	1	99	0	0	1
randu	orig	5	0	99	100	0	0	0	100
randu	orig	5	0.1	99	98	0	0	2	98
randu	g/2	5	0	76	0	0	0	100	0
randu	g/2	5	0.1	77	1	0	0	99	1
randu	g	5	0	80	0	0	0	100	0
randu	g	5	0.1	71	0	100	0	0	0
randu	2g	5	0	81	0	100	0	0	0
randu	2g	5	0.1	72	0	100	0	0	0

Notation in text

Problem of Gap Insufficiency

In “Failure of Gaps as a Criterion for Clusterability” section, we drew attention to the fact that fixing gap is an insufficient criterion to define well-clusterability for k -means like (or centre-based) algorithms. The unbalanced sample can lead to change in clustering optimum position. To confirm it, an experiment reported in Table 1 on synthetic data for a number of algorithms from k -means family and some other was performed. The k -means implementation *kmeans* from the *R* package *stats* in versions “Hartigan–Wong”, “Lloyd”, “Forgy”, “MacQueen” was used. The algorithms were run with *nstart* parameters equal 1, 10 and 20. Additionally, experiments with the function *cmeans* (implementing Fuzzy c -means algorithm) from the package *e1071* in versions “cmeans” and “ufcl” were performed. For comparison, the single link algorithm as implemented in the *hclust* function of the package *stats* was run. We implemented our own version of initialization of the k -means++ algorithm and used it in combination with the standard *kmeans* of *R*. Two variants of k -means++ were used: one with a single start and one with two starts.

We fix the gap parameter to be equal $g/r_{\max} = 2$, as, e.g. proposed in so-called perfect clusterability criterion.

With increase of the spread between cluster sizes, all but single link algorithm have increasing difficulty in detecting the “perfect” clusters. The additionally displayed indicator *relQ* (relative quality, the quotient of the achieved k -means cost function to the smallest one) explains the reason: k -means optimizes k -means clustering function and single link does not.

k -Means++ Discovering Well-Clusterable Data for Synthetic Data

In Tables 2, 3 and 4, illustrating the second and third experiment sets, the following abbreviations in column titles were used: Errors k -means means the number of errors (not recognizing the correct clustering) per 100 runs by the k -means algorithm with random initialization with 1 start. Errors k -means++ means the same, respectively. WC disc. means the number of times per 100 runs that the k -means++ discovered the well-clusterable structure in the data, whereas WC not disc. means the number of times k -means++ did not discover such a structure. Both are split into two categories: cc—correct clustering in the sense of the assumed cluster structure, wc—wrong clustering in the same sense.

The set-up of the second experiment is as follows: a generator with parameters k (number of clusters), M , m (minimum and maximum cluster size), d (dimensionality), p (share of variance outside of core) and gp (the minimum gap size compared to the one required to testify that well-clusterable structure was found). Some parameters are fixed

for this experiment: $M = 75$, $m = 45$, $d = 10$, and only k , p and gp are varied. In each run, a new sample is generated.

You see from Table 2 that k -means++ almost always discovers the well-clusterable structure in the data when it exists (it failed one time in 1800 experiments) and when it failed, the discovered structure was non-well-clusterable one. In 1800 experiments, when the well-clusterable structure did not exist, it did not discover a well-clusterable structure. It is worth noting that given the gap between the clusters is half as big as the well-clusterability gap, k -means++ discovered nearly always (except for 4 cases) the intrinsic structure in the data. Interestingly, k -means discovers the intrinsic structure in the data when $k = 2$. But it fails when the number of clusters is bigger and the increase of the gaps between clusters does not improve its performance.

k -Means++ Discovering Well-Clusterable Data for Real Data

The set-up for the third experiment is as follows. Datasets from the *R* library with at least 100 records were selected (“DNase”, “iris”, “Theoph”, “attenu”, “faithful”, “inert”, “quakes”, “randu”). Non-numeric columns were removed, the numerical columns were normalized (mean = 0, SD = 1) and a clustering via k -means++ into $k = (2, 3, 5)$ clusters with 200 starts was performed and this was considered as the “ground truth” clustering. Then 100 times clustering experiments were performed computing the statistics as in the second experiment for the original data set and for data set modified so that the gaps between clusters were equal $g/2$, g , $2g$ where g is the gap required by the well-clusterability condition.

In an analogous way, *kddcup* data downloaded from the website <https://sci2s.ugr.es/keel/category.php?cat=clas&order=ins#sub2> was processed. Not the entire set but rather randomly selected some 10,000 records from it were used. This extra data set was used to show the performance of the clustering for a larger number of clusters (2, 4, 8, 16, 32, 64 and 128). As visible, the algorithm tackles the increasing number of clusters correctly.

One sees that both k -means++ and k -means do not perform well for the original data set. But if the gaps between clusters amount to at least half the well-clusterability gap, k -means++ discovers the intrinsic clustering. k -means does so only when $k = 2$. k -means++ discovers always the well-clusterable structure when it exists and does not if does not exist. If $k > 2$, k -means is not helped by increasing the gap between clusters.

Summarizing, it has been demonstrated empirically that (1) The gap size alone does not guarantee the discovery of clusters seen by human inspection for the class of k -means-like algorithms. (2) If a dataset is well-clusterable as defined in Theorems 1 or 5 (based on Definition 1) then k -means++

Table 4 Reconstruction of the clusters in real data from the *kddcup*

Dataset	Gap type	Clusters	p	Errors		WC disc.		WC not disc.	
				<i>k</i> -means	<i>k</i> -means++	cc	wc	cc	wc
kddcup	orig	2	0	82	75	0	0	25	75
kddcup	g/2	2	0	0	0	0	0	100	0
kddcup	g	2	0	0	0	100	0	0	0
kddcup	2g	2	0	0	0	100	0	0	0
kddcup	orig	4	0	58	69	0	0	31	69
kddcup	g/2	4	0	66	0	0	0	100	0
kddcup	g	4	0	64	0	100	0	0	0
kddcup	2g	4	0	67	0	100	0	0	0
kddcup	orig	8	0	98	91	0	0	9	91
kddcup	g/2	8	0	95	0	0	0	100	0
kddcup	g	8	0	93	0	100	0	0	0
kddcup	2g	8	0	95	0	100	0	0	0
kddcup	orig	16	0	100	97	0	0	3	97
kddcup	g/2	16	0	100	0	0	0	100	0
kddcup	g	16	0	100	0	100	0	0	0
kddcup	2g	16	0	100	0	100	0	0	0
kddcup	orig	32	0	100	100	0	0	0	100
kddcup	g/2	32	0	100	0	0	0	100	0
kddcup	g	32	0	100	0	100	0	0	0
kddcup	2g	32	0	100	0	100	0	0	0
kddcup	orig	64	0	100	100	0	0	0	100
kddcup	g/2	64	0	100	0	0	0	100	0
kddcup	g	64	0	100	0	100	0	0	0
kddcup	2g	64	0	100	0	100	0	0	0
kddcup	orig	128	0	100	100	0	0	0	100
kddcup	g/2	128	0	100	0	0	0	100	0
kddcup	g	128	0	100	0	100	0	0	0
kddcup	2g	128	0	100	0	100	0	0	0

Notation in text

is able to identify the best clustering (both for real-world datasets and synthetic ones). (3) If *k*-means++ cannot find a clustering satisfying well-clusterability, there is no good clustering structure, fitting those definitions, hidden in data (with high probability) for all *k*-means style algorithms.

Impact of *k*, Data Dimensionality and Gap on Clustering Performance

A number of further experiments have been performed on synthetic data in order to show the behaviour of several algorithms from *k*-means family on data that were generated with *k*-means++ clusterability in mind.

The impact of the parameters *k* (the number of clusters), *d* (the number of dimensions), the cluster size and the quotient of the maximum to the minimum cluster size were investigated. Additionally, the impact of the reduction of the gap between clusters on the performance of the clustering was

studied (*breakgap* parameter indicates by what number the calculated clusterability gap was divided).

The impact was measured as follows: 10 different cluster sets were generated randomly for each value of the parameter (e.g. for *k* = 2, 4, 8, and 16 clusters), separated by appropriate gap. Then for 1000 times each of the algorithms was applied. Each outcome was classified as either correct or incorrect. The outcome was correct if exactly the predefined set of clusters was discovered by the run of the algorithm. Otherwise, an error was reported. Each cell in the tables contains four numbers: the average number of errors over the 10 cluster sets (number of errors out of 1000 runs), plus the standard deviation, the time in seconds taken by 1000 runs of the algorithm and the average obtained relative quality. The relative quality was measured as a quotient of the *Q* cost function value divided by the minimal one obtained by any algorithm for a given cluster set.

Both the case of gap separation of clusters (see Tables 5, 6, 7, 8, 9) and the gap separation of cores (see Tables 10,

Table 5 Dependence of the number of errors on k

$k =$	2	4	8	16
Hartigan–Wong 1 ×	0 ± 0	583 ± 158.5	938.3 ± 18.7	999.4 ± 0.8
1000 execs in s/relQ	0.24/1	0.3/701.98	0.44/54638.17	0.68/3276669.6
Lloyd 1 ×	0 ± 0	655.8 ± 78.2	952.7 ± 9.4	999.7 ± 0.5
1000 execs in s/relQ	0.23/1	0.29/1056.44	0.43/71083.66	0.66/4201145
Forgy 1 ×	0 ± 0	645 ± 75.3	950.1 ± 8.1	999.9 ± 0.3
1000 execs in s/relQ	0.23/1	0.28/1112.96	0.43/71431.91	0.66/4175279.99
MacQueen 1 ×	0 ± 0	642.5 ± 64.2	949.9 ± 14.7	999.6 ± 0.7
1000 execs in s/relQ	0.23/1	0.28/1109.83	0.43/73197.95	0.66/4131048.66
Hartigan–Wong 10 ×	0 ± 0	13 ± 10.4	540.9 ± 107.6	991.5 ± 10.7
1000 execs in s/relQ	0.58/1	0.61/7.26	0.99/4419.42	1.38/816542.71
Lloyd 10 ×	0 ± 0	18.8 ± 16.9	621.4 ± 55.9	995.2 ± 4.6
1000 execs in s/relQ	0.51/1	0.55/9.83	0.92/6123.88	1.27/1063785.95
Forgy 10 ×	0 ± 0	20.8 ± 15.4	613.9 ± 58.5	996.2 ± 2.1
1000 execs in s/relQ	0.51/1	0.55/11.67	0.92/5976.84	1.27/1059280.68
MacQueen 10 ×	0 ± 0	16.2 ± 11.9	616.8 ± 68.3	996.1 ± 4.4
1000 execs in s/relQ	0.51/1	0.53/9.28	0.87/5611.13	1.24/1044377.39
Hartigan–Wong 20 ×	0 ± 0	0 ± 0	281.8 ± 101.9	986 ± 17.2
1000 execs in s/relQ	0.76/1	0.83/1	1.28/1681.53	1.78/536122.93
Lloyd 20 ×	0 ± 0	0.4 ± 0.7	385.6 ± 66.5	992.8 ± 6.3
1000 execs in s/relQ	0.63/1	0.72/1.13	1.14/2597.6	1.58/693511.33
Forgy 20 ×	0 ± 0	0.8 ± 1	389.2 ± 62.2	992.6 ± 6.5
1000 execs in s/relQ	0.63/1	0.71/1.37	1.14/2670.53	1.58/680391.36
MacQueen 20 ×	0 ± 0	0.7 ± 0.8	381.4 ± 91.3	992.7 ± 6.3
1000 execs in s/relQ	0.63/1	0.66/1.37	1.06/2374.48	1.52/695009
cmeans Fuzzy	0 ± 0	163 ± 184.5	40 ± 63.3	35.1 ± 67.3
1000 execs in s/relQ	0.48/1	0.56/48.39	1.02/148.2	2.05/1213.9
ufcl Fuzzy	7.8 ± 6.4	725 ± 62	971.4 ± 7.5	999.8 ± 0.4
1000 execs in s/relQ	0.56/1.14	0.68/1646.94	1.51/109702.3	3.03/5599855.51
single link	0 ± 0	0 ± 0	0 ± 0	0 ± 0
1000 execs in s/relQ	0.15/1	0.18/1	0.33/1	0.56/1
k -means++	0 ± 0	2.1 ± 1.6	0.2 ± 0.4	0 ± 0
1000 execs in s/relQ	0.78/1	1.78/1.99	9.4/1.55	33.43/1
k -means++ 2 ×	0 ± 0	0 ± 0	0 ± 0	0 ± 0
1000 execs in s/relQ	1.51/1	3.53/1	18.76/1	66.45/1

Other parameters fixed at $d = 2$ $M = 15$, $m = 9$ $breakgap = 1$

11, 12, 13, 14, 15) were investigated. In the case of cores, correctness of proper classification of core elements was considered only.

Two basic insights can be gained from all these tables. The well-clusterability concept defined in this paper is well suited for the single link algorithm. Single link is designed to identify data where there are large gaps between clusters. It is also significantly quicker than k -means++. k -means++ is the second best performing algorithm, but at the same time the slowest one. The poor comparison of speed results from the fact that only small data sets were used. Single link speed grows however quadratically in the size of the data set, while k -means speed depends quadratically on k and linearly on the size of the data set. Besides, the memory consumption

of single link is quadratic in the size of the data, while that of k -means++ is linear. Hence single link is a competitor for toy examples only.

Table 5 demonstrates that two-cluster task is easy for all the algorithms except for *ufcl* variant of Fuzzy-*c*-means. However, with an increase in the number of clusters, the capability of detecting the optimal clustering deteriorates strongly except for the two mentioned algorithms. The *cmeans* variant of Fuzzy *c*-means is the least affected among algorithms 1:14. Understandably, the variants of k -means with more restarts perform better than those with smaller number of restarts. Same conclusions can be drawn from the core-based clusterability experiment summarized in

Table 6 Dependence of the number of errors on d

$d =$	2	4	8	16
Hartigan–Wong 1 ×	718.9 ± 65.5	695.8 ± 46	748.9 ± 43.2	750.8 ± 28.7
1000 execs in s/relQ	0.34/1762.09	0.42/749.98	0.59/474.19	0.92/447.61
Lloyd 1 ×	741.8 ± 59.9	721.2 ± 31.9	779.8 ± 35.2	764.7 ± 14.2
1000 execs in s/relQ	0.34/2108.83	0.41/860.33	0.57/520.88	0.9/470.34
Forgy 1 ×	742.5 ± 52.2	718.5 ± 28.7	773.6 ± 41.9	756.3 ± 16.7
1000 execs in s/relQ	0.33/2116.47	0.41/844.73	0.57/517.54	0.89/467.36
MacQueen 1 ×	747 ± 47.3	716.2 ± 38.5	765.4 ± 46.7	751.2 ± 22.3
1000 execs in s/relQ	0.34/2173.94	0.41/831	0.57/503.37	0.9/459.45
Hartigan–Wong 10 ×	45 ± 33.8	35 ± 21	61.3 ± 26.4	52.9 ± 13.6
1000 execs in s/relQ	0.77/28.8	0.92/14.86	1.28/18.92	2.11/21.8
Lloyd 10 ×	57.1 ± 37	37.2 ± 14.4	86 ± 41.6	63.4 ± 14.8
1000 execs in s/relQ	0.67/35.61	0.85/15.55	1.21/26.3	2/26.12
Forgy 10 ×	56.7 ± 35	38.7 ± 15.2	86.1 ± 47.1	63.3 ± 12.5
1000 execs in s/relQ	0.68/35.61	0.85/16.23	1.22/26.04	2.01/26.02
MacQueen 10 ×	55.4 ± 40.6	41.7 ± 17.7	85.5 ± 49.9	63.1 ± 11.6
1000 execs in s/relQ	0.67/34.85	0.82/17.41	1.18/25.78	2/25.96
Hartigan–Wong 20 ×	3.7 ± 4.1	0.8 ± 1.3	3.5 ± 3.5	4.2 ± 2.8
1000 execs in s/relQ	1/3.29	1.16/1.33	1.54/1.98	2.42/2.63
Lloyd 20 ×	4.1 ± 4	1.9 ± 1.8	10.1 ± 9.5	3.5 ± 1.7
1000 execs in s/relQ	0.84/3.28	1.03/1.73	1.42/3.79	2.25/2.41
Forgy 20 ×	4.4 ± 6.9	2 ± 1.9	9.9 ± 12.3	4.3 ± 2.5
1000 execs in s/relQ	0.84/3.41	1.03/1.75	1.42/3.68	2.25/2.67
MacQueen 20 ×	4.8 ± 5.4	2.3 ± 2.5	8.6 ± 9.2	3.2 ± 1.9
1000 execs in s/relQ	0.82/3.85	0.98/1.83	1.36/3.42	2.21/2.27
cmeans Fuzzy	155 ± 159.3	16 ± 35.3	19.2 ± 60.7	0 ± 0
1000 execs in s/relQ	0.71/101.05	0.78/7.21	0.95/6.14	1.33/1
ufcl Fuzzy	800.7 ± 37.8	797.8 ± 28.4	867.8 ± 21.9	883.5 ± 20.3
1000 execs in s/relQ	0.92/5952.77	1.1/1097.38	1.57/649.79	2.69/522.86
single link	0 ± 0	0 ± 0	0 ± 0	0 ± 0
1000 execs in s/relQ	0.24/1	0.29/1	0.43/1	0.7/1
k-means++	1.2 ± 0.9	1.9 ± 1.9	4.8 ± 3.6	2.7 ± 1.7
1000 execs in s/relQ	3.45/1.75	3.34/2.06	3.69/2.54	4.52/2.17
k-means++ 2 ×	0 ± 0	0 ± 0	0 ± 0	0 ± 0
1000 execs in s/relQ	6.83/1	6.58/1	7.09/1	8.53/1

Other parameters fixed at $k = 5$ $M = 15$, $m = 9$ $breakgap = 1$

Table 10, but of course the off-core elements contribute to worsening of the performance.

The impact of dimensionality is shown in Tables 6 and 11. The increase of dimensionality slightly deteriorates the performance (compared to the impact of the number of clusters), and in case of *cmeans* it seems to improve it.

Tables 8 and 13 present the influence of the cluster sizes on the performance. The increase of cluster size negatively influences the performance except for *cmeans* which seems to take advantage of the increased sample size.

The spread of sample sizes, as illustrated by Tables 7 and 12 apparently does not affect the performance. (Though again *cmeans* takes advantage of it in the case represented

by Table 7.) This would mean that the spread is quite well captured by our formula on g .

The effect of decreasing the gap size below the value determined by our formulas, as shown in Tables 9 and 14, surprisingly worsens the performance of *cmeans* and *k-means++* only. Traditional *k-means* algorithms with 20 restarts outperform *k-means++* apparently. This would mean that *k-means++* loses its advantage of high probability of hitting distant points in case of small gaps because these distant points may be points within the same cluster rather than points of distinct clusters.

As visible from Table 15, the share of off-core variance seems not to affect the performance of the algorithm, except for deteriorating *cmeans* performance.

Table 7 Dependence of the number of errors on M to m proportion

$M, m =$	15, 9	30, 9	60, 9	120, 9
Hartigan–Wong 1 ×	723.7 ± 43.5	801 ± 58.1	762.9 ± 49.4	736.1 ± 76.4
1000 execs in s/relQ	0.36/6216.55	0.37/10441.43	0.41/2427.55	0.45/5018.41
Lloyd 1 ×	730.4 ± 50.7	804.9 ± 67.4	777.3 ± 54.6	733.8 ± 88.3
1000 execs in s/relQ	0.35/7085.08	0.36/11054.14	0.4/2456.29	0.45/5112.28
Forgy 1 ×	731.3 ± 50.2	809.8 ± 67	775.2 ± 57.9	748.9 ± 81
1000 execs in s/relQ	0.34/7404.61	0.36/10798.16	0.41/2508.64	0.45/5033.54
MacQueen 1 ×	726.8 ± 44.9	809 ± 70.7	777.7 ± 53	741.3 ± 88.3
1000 execs in s/relQ	0.35/7206.3	0.35/11617.69	0.39/2508.61	0.43/4884.21
Hartigan–Wong 10 ×	44.4 ± 26.7	126.3 ± 73.5	83.8 ± 52	69 ± 47.1
1000 execs in s/relQ	1.1/24.72	1.32/119.48	2.56/59.89	3.79/47.41
Lloyd 10 ×	46.6 ± 29.8	138.8 ± 88.7	93.2 ± 55.9	75.2 ± 51.4
1000 execs in s/relQ	1.01/25.78	1.23/114.56	2.57/63.23	3.83/51.64
Forgy 10 ×	48.7 ± 34.1	143.5 ± 91.1	95 ± 56.1	69.9 ± 49.1
1000 execs in s/relQ	1.01/27.43	1.22/149.46	2.56/64.55	3.82/48.17
MacQueen 10 ×	46.5 ± 25.4	138.6 ± 87.6	82.5 ± 52.4	66.9 ± 46.8
1000 execs in s/relQ	0.99/26.53	1.19/142.97	2.38/57.64	3.57/46.12
Hartigan–Wong 20 ×	2.2 ± 3.6	17.2 ± 15.7	8.7 ± 11.2	6.3 ± 6.8
1000 execs in s/relQ	1.38/2.15	1.63/9.17	3.04/6.78	4.45/4.91
Lloyd 20 ×	3.5 ± 4.4	26.3 ± 27.5	11 ± 13.7	8.6 ± 10
1000 execs in s/relQ	1.19/2.81	1.45/16.83	3.08/7.88	4.58/6.92
Forgy 20 ×	3.2 ± 4	27.7 ± 28.1	11.2 ± 10.3	7.9 ± 8.4
1000 execs in s/relQ	1.19/2.62	1.45/13.49	3.08/7.91	4.57/6.1
MacQueen 20 ×	3.8 ± 6.1	23.7 ± 24.6	9.8 ± 12.4	7.4 ± 8.1
1000 execs in s/relQ	1.17/2.89	1.4/11.74	2.72/7.28	4.07/5.74
cmeans Fuzzy	100 ± 120.3	119.2 ± 113.8	46.9 ± 51.9	59 ± 83
1000 execs in s/relQ	0.99/51.88	1.15/64.07	2.09/35.43	3.07/48.86
ufcl Fuzzy	810.6 ± 19.5	866.6 ± 23.3	828.8 ± 31.9	846.4 ± 25.1
1000 execs in s/relQ	1.48/15995.33	1.83/14805.01	3.88/3690.75	5.91/14135.23
single link	0 ± 0	0 ± 0	0 ± 0	0 ± 0
1000 execs in s/relQ	0.34/1	0.44/1	1.47/1	3.4/1
k -means++	1.3 ± 1.1	1.3 ± 1.1	1.4 ± 1.2	1.1 ± 1.4
1000 execs in s/relQ	7.25/1.7	9.59/1.63	23.52/2.69	37.31/1.69
k -means++ 2 ×	0 ± 0	0 ± 0	0 ± 0	0 ± 0
1000 execs in s/relQ	14.48/1	19.12/1	47.03/1	74.88/1

Other parameters fixed at $k = 5$ $d = 2$ $breakgap = 1$

Discussion

Ben-David [10] investigated several clusterability notions under the following criteria:

- practical relevance—most real-world data sets should be clusterable;
- existence of quick clustering algorithms
- existence of quick clusterability testing algorithms
- some popular algorithms should perform well on clusterable data.

As the practical relevance is concerned, Ben-David found that the clusterability criteria he investigated are

usually not met because of the impractically large gap between cluster centres is required. Our proposed clusterability criteria are as impractical as the ones he investigated—large gaps are required and their size grows with the spread of cluster sizes, the number of clusters, the cluster radius. This is due among others to the fact that k -means does not explore directly the gaps between clusters. With this respect, our proposal does not constitute a progress.

As the algorithm complexity is concerned, k -means++ has no computational disadvantage over the algorithms Ben-David considered. The gain one has is that usually one iteration of k -means is needed to establish the final result (the whole complexity resides in the initialization stage).

Table 8 Dependence of the number of errors on cluster size, M and m kept proportional

$M, m =$	15, 9	30, 18	60, 36	120, 72
Hartigan–Wong 1 ×	714.5 ± 55.3	726.7 ± 41	751.6 ± 35.1	739.4 ± 48.6
1000 execs in s/relQ	0.36/2222.82	0.38/9506.9	0.4/2452.25	0.48/1017.77
Lloyd 1 ×	731.6 ± 39.9	737.9 ± 34.2	743.6 ± 42.8	733.5 ± 34.2
1000 execs in s/relQ	0.35/2542.88	0.37/9537.46	0.4/2376.59	0.48/1011.3
Forgy 1 ×	725.9 ± 46.8	734.9 ± 49	756.1 ± 39	739.2 ± 36.8
1000 execs in s/relQ	0.35/2599.24	0.37/9968.81	0.39/2606.82	0.48/1007.41
MacQueen 1 ×	736.8 ± 41.3	735.2 ± 39.8	753 ± 42.5	734.5 ± 35.1
1000 execs in s/relQ	0.35/2547.24	0.36/9369.12	0.38/2509.95	0.45/970.99
Hartigan–Wong 10 ×	41.8 ± 25.7	48.1 ± 29.8	61.1 ± 29.7	54.1 ± 32.9
1000 execs in s/relQ	1.05/37.25	1.84/28.47	2.36/39.86	4.83/19.06
Lloyd 10 ×	47.3 ± 32.4	53.7 ± 27.2	64.8 ± 31.4	53.8 ± 30.3
1000 execs in s/relQ	0.97/40.99	1.63/31.44	2.28/42.89	4.9/18.99
Forgy 10 ×	52.6 ± 33.2	49.3 ± 26.7	64.6 ± 27.4	51.3 ± 25.2
1000 execs in s/relQ	0.96/45.82	1.63/29.53	2.28/42.94	4.91/18.29
MacQueen 10 ×	50.2 ± 31.2	50.5 ± 27	65.2 ± 28.9	54 ± 29.2
1000 execs in s/relQ	0.95/44.05	1.57/29.42	2.19/42.68	4.59/19.07
Hartigan–Wong 20 ×	2.7 ± 3.5	3.3 ± 3.3	3.7 ± 2.8	2.5 ± 2.3
1000 execs in s/relQ	1.33/3.24	2.07/2.84	2.81/3.58	5.64/1.82
Lloyd 20 ×	4.5 ± 4.9	2.9 ± 3	5 ± 3.9	2.3 ± 2.9
1000 execs in s/relQ	1.16/4.61	1.91/2.61	2.67/4.13	5.81/1.78
Forgy 20 ×	4 ± 4.5	3 ± 3	4.6 ± 3.9	3.1 ± 3.1
1000 execs in s/relQ	1.16/4.22	1.91/2.66	2.66/3.87	5.82/2.06
MacQueen 20 ×	2.7 ± 3.3	2.9 ± 2.9	3.8 ± 3.8	2.3 ± 2.5
1000 execs in s/relQ	1.11/3.2	1.81/2.61	2.51/3.33	5.2/1.75
cmeans Fuzzy	23.9 ± 33.9	70.7 ± 117.4	82.4 ± 87.2	55.3 ± 50.3
1000 execs in s/relQ	0.92/22.94	1.45/39.24	2/62.64	3.94/18.83
ufcl Fuzzy	813.5 ± 24.8	808.6 ± 15.9	805.2 ± 18.5	825.3 ± 20.2
1000 execs in s/relQ	1.39/4980.44	2.5/12524.72	3.57/5516.03	7.71/1798.34
single link	0 ± 0	0 ± 0	0 ± 0	0 ± 0
1000 execs in s/relQ	0.32/1	0.68/1	1.27/1	6.13/1
k -means++	0.8 ± 1.3	0.6 ± 0.8	0.6 ± 0.7	1.9 ± 1.3
1000 execs in s/relQ	6.57/1.61	14.13/1.32	21.48/1.43	49.4/1.73
k -means++ 2 ×	0 ± 0	0 ± 0	0 ± 0	0 ± 0
1000 execs in s/relQ	13.09/1	28.24/1	42.93/1	98.78/1

Other parameters fixed at $k = 5$ $d = 2$ $breakgap = 1$

With respect to efficient testability of the clusterability conditions Ben-David complains about the problem that checking clusterability condition requires knowledge of the optimal clustering that is NP-hard to find. With our proposal, we make here a major progress. Given an output clustering, if one checks that the distances between clusters match the requirements imposed on the gap g , one is assured that one has found the optimal clustering. On the other hand, if the data is clusterable in our sense, then it is highly unlikely that the standard k -means++ algorithm does not find it.

With respect to the last requirement, one can say that k -means++ is actually a popular algorithm and it behaves reasonably.

Clusterable Data Sets for Varying Values of k

The described clusterability criteria are based on the knowledge of the k value to check for. The following questions can be raised in this context:

- Does there exist some another k for which the clusterability criterion holds?
- Can one “guess” beforehand what k shall one check in order to check for potential clusterability?

Let us restrain to non-core cases only here.

Table 9 Dependence of the number of errors on *breakgap*

<i>breakgap</i> =	1	2	4	8
Hartigan–Wong 1 ×	727.8 ± 47.5	713.8 ± 35.1	697.3 ± 69.2	663.3 ± 42.7
1000 execs in s/relQ	0.36/1312.52	0.36/710.88	0.37/133.68	0.36/55.72
Lloyd 1 ×	735.7 ± 42.4	728 ± 35.7	717.7 ± 29.6	690.8 ± 32.5
1000 execs in s/relQ	0.35/1469.84	0.35/720.23	0.37/143.39	0.36/61.49
Forgy 1 ×	733.1 ± 39.7	729.8 ± 27.5	721.2 ± 39.9	689.2 ± 22.3
1000 execs in s/relQ	0.35/1421.55	0.34/715.94	0.37/145.41	0.35/63.16
MacQueen 1 ×	736.1 ± 43.4	717.4 ± 35.9	711.9 ± 51.5	684.9 ± 29.1
1000 execs in s/relQ	0.35/1440.8	0.34/764.85	0.36/140.76	0.34/63.32
Hartigan–Wong 10 ×	52.6 ± 25.3	36.7 ± 14.8	31.4 ± 21.2	20.9 ± 12.3
1000 execs in s/relQ	1.01/22.4	0.98/5.98	1.06/2.18	1/1.23
Lloyd 10 ×	49.5 ± 22	39.7 ± 17.1	41.4 ± 25.1	24.8 ± 4.7
1000 execs in s/relQ	0.92/21.14	0.91/5.92	0.99/2.62	0.96/1.29
Forgy 10 ×	51.4 ± 22.1	42.6 ± 14.9	42.3 ± 20.6	26.6 ± 8.9
1000 execs in s/relQ	0.93/22.14	0.91/6.52	1/2.65	0.96/1.3
MacQueen 10 ×	50.4 ± 25.9	39.2 ± 17.6	40.9 ± 20.5	20.7 ± 6.9
1000 execs in s/relQ	0.9/21.57	0.88/5.86	0.95/2.61	0.89/1.24
Hartigan–Wong 20 ×	3.4 ± 2.6	1.6 ± 1.9	1.7 ± 2.9	0.9 ± 1.5
1000 execs in s/relQ	1.27/2.32	1.25/1.19	1.35/1.06	1.27/1.01
Lloyd 20 ×	2.8 ± 2.7	1.8 ± 1.5	1.7 ± 2.2	0.7 ± 0.8
1000 execs in s/relQ	1.11/2.09	1.11/1.23	1.22/1.06	1.21/1.01
Forgy 20 ×	3.1 ± 2.4	3.4 ± 1.6	2.8 ± 3.2	0.9 ± 0.9
1000 execs in s/relQ	1.11/2.19	1.12/1.42	1.21/1.11	1.21/1.01
MacQueen 20 ×	2.4 ± 2.5	2.2 ± 2.5	1.1 ± 1.9	0.3 ± 0.5
1000 execs in s/relQ	1.07/1.92	1.05/1.26	1.13/1.04	1.06/1
cmeans Fuzzy	57.2 ± 67.9	159.8 ± 153.7	171.5 ± 109.1	247.8 ± 128.8
1000 execs in s/relQ	0.89/25.26	0.9/26.38	0.98/7.94	0.93/4.18
ufcl Fuzzy	800.8 ± 29.2	801.4 ± 16.7	853.2 ± 34.1	901.3 ± 36.2
1000 execs in s/relQ	1.32/1989.2	1.3/1645.95	1.4/289.28	1.31/105.34
single link	0 ± 0	0 ± 0	0 ± 0	0 ± 0
1000 execs in s/relQ	0.3/1	0.3/1	0.32/1	0.3/1
k-means++	1.6 ± 1.1	4.9 ± 3.2	12.1 ± 6.5	47.5 ± 19.4
1000 execs in s/relQ	6.1/1.68	5.95/1.66	6.54/1.49	6.03/1.57
k-means++ 2 ×	0 ± 0	0 ± 0	0 ± 0	3.2 ± 2.9
1000 execs in s/relQ	12.12/1	11.82/1	13.08/1	11.96/1.03

Other parameters fixed at $k = 5$ $d = 2$ $M = 15$, $m = 9$

The first question has one trivial and some non-trivial answers. If k is equal to sample size, then the clusterability condition is trivially fulfilled.

But given a data set is clusterable for k , what can one say about clusterability for $k + k'$ ($k' \geq 1$)? Assume that the optimal clustering for $k + k'$, $\mathcal{C}^{(k+k')}$, engages such clusters $C_j^{(k+k')}$, $j = 1, \dots, k + k'$ that each contains objects from different clusters from the optimal clustering for k , $\mathcal{C}^{(k)}$. Then one can have the case that cluster centres of clusters $C_j^{(k+k')}$ are for each element in the cluster $C_i^{(k)}$ of k clustering at least $r_i^{(k)} = r_i$ away from the border of the ball enclosing cluster $C_i^{(k)}$. This would be impossible because in such a

case clustering $\mathcal{C}^{(k)}$ would have a cost function lower than the clustering $\mathcal{C}^{(k+k')}$. Therefore, consider a cluster $C_j^{(k+k')}$ with centre at the distance at least r_i and at most $2r_i$ from the cluster $C_i^{(k)}$. It has to have a radius of at least $g - r_i > r_i$, where g is the gap for k clusterability of the data set. Each element of the cluster $C_i^{(k)}$ is then at most $3r_i$ away from the centre of $C_j^{(k+k')}$. The required gap g' for $k + k'$ clusterability would amount to more than $2(g - r_i) > 2r_i$. Which means that all elements of the cluster $C_i^{(k)}$ must belong to $C_j^{(k+k')}$. Under these circumstances only $k - 1$ clusters of the clustering $\mathcal{C}^{(k+k')}$ are non-empty which is a contradiction. If one of the clusters of $\mathcal{C}^{(k+k')}$ is identical with one from $\mathcal{C}^{(k)}$, then one can drop it and reason as above.

Table 10 Dependence of the number of errors on k

$k =$	2	4	8	16
Hartigan–Wong 1 ×	0 ± 0	531 ± 93.8	941.3 ± 11.5	999.3 ± 0.8
1000 execs in s/relQ	0.25/1	0.33/1484.76	0.49/494624.56	0.81/52872221.3
Lloyd 1 ×	0 ± 0	520.8 ± 97.7	942.3 ± 11.7	999.4 ± 0.7
1000 execs in s/relQ	0.24/1	0.32/1492.16	0.48/502271.64	0.8/58530442.16
Forgy 1 ×	0 ± 0	524.9 ± 81.9	940 ± 12.2	998.8 ± 1.1
1000 execs in s/relQ	0.24/1	0.32/1548	0.48/520427.63	0.8/54534626.71
MacQueen 1 ×	0 ± 0	520.5 ± 82.6	945 ± 13.2	999.4 ± 0.5
1000 execs in s/relQ	0.24/1	0.32/1495.12	0.47/532067.28	0.79/57986428.7
Hartigan–Wong 10 ×	0 ± 0	2.6 ± 2.8	541 ± 62.9	992.9 ± 2.6
1000 execs in s/relQ	0.63/1	0.93/3.41	1.54/16469.01	2.86/7858388.63
Lloyd 10 ×	0 ± 0	3.3 ± 3.1	563.2 ± 68.1	994.1 ± 3
1000 execs in s/relQ	0.56/1	0.84/4.16	1.44/18712.34	2.77/8511396.87
Forgy 10 ×	0 ± 0	3.4 ± 3	558.3 ± 71.5	993.5 ± 3.5
1000 execs in s/relQ	0.56/1	0.84/4.2	1.44/17368.35	2.76/8325871.74
MacQueen 10 ×	0 ± 0	2.9 ± 2.5	551.2 ± 59.2	992.9 ± 2.7
1000 execs in s/relQ	0.56/1	0.83/3.82	1.4/17832.1	2.66/8382457.42
Hartigan–Wong 20 ×	0 ± 0	0 ± 0	299.9 ± 66.5	986.3 ± 3.9
1000 execs in s/relQ	0.84/1	1.18/1	1.94/4916.09	3.61/4501700.03
Lloyd 20 ×	0 ± 0	0 ± 0	322.2 ± 69.9	987 ± 4.4
1000 execs in s/relQ	0.69/1	1.01/1	1.74/5325.33	3.43/4989161.44
Forgy 20 ×	0 ± 0	0.1 ± 0.3	318.9 ± 69.3	987.4 ± 4.7
1000 execs in s/relQ	0.69/1	1/1.09	1.74/5318.02	3.43/4869931.56
MacQueen 20 ×	0 ± 0	0 ± 0	308.4 ± 68.3	986.3 ± 3.8
1000 execs in s/relQ	0.69/1	0.99/1	1.65/5183.37	3.27/4810210.65
cmeans Fuzzy	0 ± 0	32.6 ± 43.8	64.3 ± 74.2	53.7 ± 76.1
1000 execs in s/relQ	0.51/1	0.78/36.04	1.67/857.64	5.06/14553.66
ufcl Fuzzy	10.3 ± 8.1	637.2 ± 22.1	958.8 ± 8.6	999.4 ± 1
1000 execs in s/relQ	0.61/1.27	1.1/3550.22	2.78/899036.18	8.83/104163506.23
single link	0 ± 0	0 ± 0	0 ± 0	0 ± 0
1000 execs in s/relQ	0.16/1	0.27/1	0.57/1	1.48/1
k -means++	0 ± 0	0.6 ± 0.7	0.2 ± 0.4	0 ± 0
1000 execs in s/relQ	0.89/1	4.1/1.81	21.14/3.59	115.57/1
k -means++ 2 ×	0 ± 0	0 ± 0	0 ± 0	0 ± 0
1000 execs in s/relQ	1.77/1	8.19/1	42.03/1	230.99/1

Other parameters fixed at $d = 2M = 15$, $m = 9$ $breakgap = 1$ $p = 0.1$

Hence, if a data set is k and $k + k'$ clusterable, then each of the clusters in $\mathcal{C}^{(k+k')}$ must be contained in one of the clusters from $\mathcal{C}^{(k)}$.

Furthermore, the cluster with the largest radius needs to be split for sure.

Now consider a $k + k'$ -means++ clustering where the data set is k -clusterable. As each of the clusters $\mathcal{C}^{(k+k')}$ is a subset of one of the clusters from $\mathcal{C}^{(k)}$, the smallest distance between clusters in $\mathcal{C}^{(k+k')}$ would be for sure smaller than the required gap g for $\mathcal{C}^{(k)}$ clusters. Hence seeking for $k < K$ clusterability could be run as follows. Perform K -means++. Check for clusterability. remove one of the centres of two closest clusters ($K := K - 1$). Recluster

with seeds being the remaining centres. And repeat the procedure till clusterability is found or $K = 1$.

Clusterability Criteria for Other Clustering Algorithms and Cluster Models

Algorithms from the k -means family are known to work well for Gaussian-like data, where clusters are generated based on compactness. However, for data distributed differently (e.g., concentric rings), k -means usually fails. Therefore, in such situations, clustering algorithms, such as spectral clustering or kernel function based clustering are claimed to better handle such data by leveraging the connectivity information.

Table 11 Dependence of the number of errors on d

$d =$	2	4	8	16
Hartigan–Wong 1 ×	707.1 ± 50.5	744.6 ± 38.5	753.3 ± 21.4	764.7 ± 19.4
1000 execs in s/relQ	0.36/8483.66	0.45/3683.63	0.62/1761.1	0.97/2022.89
Lloyd 1 ×	717 ± 52.5	760.8 ± 39.9	753.4 ± 18.8	770.1 ± 16.8
1000 execs in s/relQ	0.35/8896.63	0.43/3873.14	0.61/1802.6	0.95/2088.42
Forgy 1 ×	711.6 ± 53.1	755.1 ± 36.5	756 ± 16.9	765.2 ± 25.3
1000 execs in s/relQ	0.35/8982.63	0.43/3741.16	0.61/1795.73	0.95/2105.67
MacQueen 1 ×	720.5 ± 44.3	753 ± 35.8	748.9 ± 20.4	769.5 ± 25
1000 execs in s/relQ	0.35/8986.52	0.43/3875.21	0.61/1797.93	0.95/2104.26
Hartigan–Wong 10 ×	38.9 ± 20.8	53.8 ± 24.6	59.8 ± 13.3	70.7 ± 23.2
1000 execs in s/relQ	1.07/87.64	1.4/67.03	2.02/76.71	3.27/106.07
Lloyd 10 ×	48 ± 23.2	65.4 ± 25.9	62.8 ± 12.6	81.5 ± 24.7
1000 execs in s/relQ	0.97/106.61	1.31/81.6	1.92/81.15	3.17/122.66
Forgy 10 ×	43.9 ± 21.5	68.1 ± 26.1	64.2 ± 12.5	82.4 ± 20.5
1000 execs in s/relQ	0.97/98.81	1.3/85.49	1.91/82.4	3.16/124.99
MacQueen 10 ×	43.6 ± 20.8	64.6 ± 23.6	64.5 ± 12.1	80.5 ± 23.7
1000 execs in s/relQ	1.35/99.1	1.29/81.23	1.89/83.85	3.12/121.61
Hartigan–Wong 20 ×	2.3 ± 2.4	3.3 ± 2.7	4.2 ± 3.3	5.1 ± 3.3
1000 execs in s/relQ	1.35/5.9	1.71/4.99	2.37/6.34	3.72/8.42
Lloyd 20 ×	2.9 ± 2.7	4.2 ± 3.5	4.3 ± 2.9	6.1 ± 3.1
1000 execs in s/relQ	1.15/7.34	1.51/6.08	2.17/6.42	3.52/10.17
Forgy 20 ×	1.6 ± 1.6	5.9 ± 3.9	3.2 ± 0.9	7.5 ± 3.1
1000 execs in s/relQ	1.15/4.41	1.51/8.15	2.17/4.98	3.51/12.02
MacQueen 20 ×	3.4 ± 2.6	5.7 ± 3.2	2.9 ± 1.7	8.2 ± 5.3
1000 execs in s/relQ	1.13/8.21	1.48/7.92	2.13/4.57	3.43/12.97
cmeans Fuzzy	41.5 ± 65.6	10.6 ± 18	0 ± 0	0 ± 0
1000 execs in s/relQ	0.97/92.62	1.04/14.1	1.21/1	1.58/1
ufcl Fuzzy	784.6 ± 20.6	805.6 ± 15.7	800.3 ± 19.4	814.9 ± 13.6
1000 execs in s/relQ	1.46/27141.38	1.86/4984.43	2.66/1940.95	4.39/2223.44
single link	0 ± 0	0 ± 0	0 ± 0	0 ± 0
1000 execs in s/relQ	0.33/1	0.41/1	0.56/1	0.87/1
k -means++	0.4 ± 0.7	0.5 ± 0.7	0.7 ± 0.8	0.5 ± 0.7
1000 execs in s/relQ	6.89/2	7.21/1.64	7.44/1.96	7.98/1.8
k -means++ 2 ×	0 ± 0	0 ± 0	0 ± 0	0 ± 0
1000 execs in s/relQ	14.03/1	14.3/1	14.62/1	15.45/1

Other parameters fixed at $k = 5$ $M = 15$, $m = 9$ $breakgap = 1$ $p = 0.1$

This picture needs to be modified slightly. Notably, the k -means underpins also several spectral clustering algorithms like ones based on the combinatorial Laplacians, normalised Laplacians and random-walk Laplacians. There exist also kernel-based variants of k -means algorithms. Therefore, the results about cluster separation presented in this research will apply in those cases also. This means that the questions asked about easiness of clustering for spectral methods may be studied in a similar way as discussed in this paper, but not in the original data space but rather in the spectral space (spanned by eigenvectors assigned to lowest eigenvalues). Of course, an investigation would be necessary into the issue how the well-separatedness in the spectral space would transpose to properties of the clusters in the

original space. The very same applies to kernel- k -means, where the clustering in the feature space follows the paradigm of classical k -means, so that the results of this paper affect the feature space. Again, a study of the various kernel functions that are typically applied, would be necessary in order to determine the relationship between well-separatedness in the feature space and the cluster shapes and/or separation in the original space.

Note also, that, as shown in this paper (see “[Failure of Gaps as a Criterion for Clusterability](#)” section), the claim that k -means works well for Gaussian distributions needs also to be verified in case of unbalanced clusters. Visually separated clusters can be clumped together and a visually uniform cluster may be split into parts already by the

Table 12 Dependence of the number of errors on M to m proportion

$M, m =$	15, 9	30, 9	60, 9	120, 9
Hartigan–Wong 1 ×	709.6 ± 40.7	778.2 ± 20.9	789 ± 36.6	765.3 ± 42.5
1000 execs in s/relQ	0.52/9331.26	0.51/9154.2	0.56/11397.3	0.68/58999.38
Lloyd 1 ×	730.6 ± 38.3	781.8 ± 19.6	786.1 ± 35.1	780.7 ± 49.8
1000 execs in s/relQ	0.5/10181.63	0.49/9648.81	0.54/11275.24	0.67/58343.78
Forgy 1 ×	723.8 ± 38.2	786.5 ± 28	792.6 ± 37	778 ± 42.6
1000 execs in s/relQ	0.49/10529.17	0.49/9641	0.55/11645.35	0.68/58111.99
MacQueen 1 ×	733.6 ± 32.4	780.2 ± 24.7	784.3 ± 38.6	767.7 ± 49.8
1000 execs in s/relQ	0.49/10804.58	0.48/10006.28	0.53/11472.56	0.63/54103.2
Hartigan–Wong 10 ×	38.3 ± 23.1	87.6 ± 28.5	96.1 ± 57.6	89.6 ± 44.1
1000 execs in s/relQ	1.63/81.12	2.15/239.35	3.04/500.28	6.44/602.73
Lloyd 10 ×	48.1 ± 25.7	87.2 ± 25	96.7 ± 46.4	79.1 ± 40.7
1000 execs in s/relQ	1.47/105.41	2.01/239.14	2.93/509.78	6.5/582.16
Forgy 10 ×	47.9 ± 19.8	86 ± 25.5	102.1 ± 45.6	89.9 ± 40.4
1000 execs in s/relQ	1.46/104.24	2/233.97	2.93/546.54	6.54/653.92
MacQueen 10 ×	46.6 ± 22.5	95.3 ± 26.4	100.8 ± 58.3	87 ± 45.5
1000 execs in s/relQ	1.42/102.86	1.96/259.02	2.8/522.96	6.13/556.78
Hartigan–Wong 20 ×	1.6 ± 3.1	8.2 ± 5.3	12.7 ± 15	7.7 ± 6.8
1000 execs in s/relQ	1.99/4.26	2.62/23.24	3.62/65.01	7.55/35.06
Lloyd 20 ×	2.7 ± 3.5	7 ± 4.1	13.3 ± 15.1	10 ± 8.3
1000 execs in s/relQ	1.74/6.54	2.35/19.65	3.41/67.85	7.72/59.9
Forgy 20 ×	2.6 ± 3.3	7.6 ± 4.1	12.7 ± 14.5	8.2 ± 5.9
1000 execs in s/relQ	1.74/6.32	2.34/21.5	3.42/66.49	7.81/39.85
MacQueen 20 ×	3.2 ± 2.9	8.2 ± 5.6	12.9 ± 14.1	9.2 ± 7.1
1000 execs in s/relQ	1.71/7.68	2.27/22.95	3.23/66.18	6.96/42.72
cmeans Fuzzy	43.1 ± 85.3	29.8 ± 59.5	4.8 ± 15.2	29.9 ± 54.3
1000 execs in s/relQ	1.42/89.96	1.84/66.83	2.52/23.17	5.23/164.93
ufcl Fuzzy	786.1 ± 26.5	815.4 ± 24.5	825.9 ± 32.1	830 ± 33.3
1000 execs in s/relQ	2.25/22096.93	3.22/13196.22	4.73/17181.09	10.65/83971.71
single link	0 ± 0	0 ± 0	0 ± 0	0 ± 0
1000 execs in s/relQ	0.49/1	0.82/1	1.48/1	7.58/1
k -means++	0.6 ± 0.7	0 ± 0	0 ± 0	0 ± 0
1000 execs in s/relQ	10.72/2.4	17.33/1	27.18/1	65.24/1
k -means++ 2 ×	0 ± 0	0 ± 0	0 ± 0	0 ± 0
1000 execs in s/relQ	21.64/1	34.62/1	54.08/1	129.98/1

Other parameters fixed at $k = 5$ $d = 2$ $breakgap = 1$ $p = 0.1$

k -means cost function. Furthermore, with high probability, the k -means algorithm with random initialization will not find the intrinsic clustering even if the clusters are well separated and the cost function points at the clustering into well-separated clusters (see “[Experimental Results](#)” section).

Conclusions

We have defined the notion of a well-clusterable data set from the point of view of the objective of k -means clustering algorithm and common sense in two variants—without any data points in the large gaps between clusters and with data points there. The novelty introduced here, compared to other work on this topic, is that one can a posteriori (after running k -means) check if the data set is well-clusterable or not.

Table 13 Dependence of the number of errors on cluster size, M and m kept proportional

$M, m =$	15, 9	30, 18	60, 36	120, 72
Hartigan–Wong 1 ×	739.4 ± 28.1	741 ± 44	745.7 ± 70	713 ± 64.6
1000 execs in s/relQ	0.37/42678.96	0.38/20676.98	0.42/5466.12	0.49/138752.72
Lloyd 1 ×	734.4 ± 28.5	741.4 ± 48.9	743.5 ± 69.7	710.4 ± 70.6
1000 execs in s/relQ	0.35/46792.33	0.37/20423.87	0.41/5537.44	0.49/151687.77
Forgy 1 ×	734.1 ± 34.3	745 ± 41.4	751.3 ± 74.9	713.5 ± 69.7
1000 execs in s/relQ	0.35/46112.12	0.36/20150.75	0.41/5548.93	0.49/144663.49
MacQueen 1 ×	738.8 ± 33.8	739.2 ± 49.6	743.6 ± 77.5	710.9 ± 66.3
1000 execs in s/relQ	0.35/45493.36	0.36/19576.95	0.4/5587.02	0.46/137172.51
Hartigan–Wong 10 ×	49 ± 19.1	57 ± 24.6	59.9 ± 21.6	48.6 ± 33.5
1000 execs in s/relQ	1.12/145.96	1.65/191.09	2.8/192.09	5/122.97
Lloyd 10 ×	54.2 ± 22.3	55.4 ± 29	64.7 ± 22.2	41.3 ± 30.7
1000 execs in s/relQ	1.02/157.45	1.55/186.28	2.75/206.05	5.02/103.98
Forgy 10 ×	56 ± 24.3	59.6 ± 27.2	66.9 ± 26.2	43.9 ± 36.1
1000 execs in s/relQ	1.02/205.13	1.55/204.96	2.74/214.45	5.02/113.15
MacQueen 10 ×	52.1 ± 21.9	57.2 ± 28.4	62 ± 27.3	42.2 ± 29.9
1000 execs in s/relQ	1.01/147.45	1.51/190.38	2.61/199.76	4.76/106.24
Hartigan–Wong 20 ×	2.8 ± 3.2	3.4 ± 2.5	4.8 ± 3.5	2.8 ± 3
1000 execs in s/relQ	1.4/8.73	1.99/12.17	3.3/16.17	5.82/7.98
Lloyd 20 ×	3 ± 2.2	4.8 ± 5.8	5.5 ± 2.6	3.5 ± 4.7
1000 execs in s/relQ	1.22/9.45	1.81/17.56	3.23/18.34	5.89/9.85
Forgy 20 ×	3 ± 3	5.6 ± 6.4	5.2 ± 3.2	2.4 ± 3.5
1000 execs in s/relQ	1.22/9.35	1.81/20.08	3.22/17.34	5.9/6.94
MacQueen 20 ×	3.3 ± 3.9	3.4 ± 2.6	4.8 ± 2.3	2.9 ± 3.7
1000 execs in s/relQ	1.19/10.02	1.74/12.45	2.98/16.03	5.39/8.23
cmeans Fuzzy	89.6 ± 106.4	46.4 ± 112.3	19 ± 36.1	122.4 ± 127.9
1000 execs in s/relQ	0.98/250.43	1.38/136.32	2.22/58.89	4.13/313.25
ufcl Fuzzy	786 ± 20.1	806.5 ± 21.9	795.7 ± 20.3	801.8 ± 33
1000 execs in s/relQ	1.51/77257.56	2.38/33556.99	4.29/16111.26	8/232714.84
single link	0 ± 0	0 ± 0	0 ± 0	0 ± 0
1000 execs in s/relQ	0.35/1	0.66/1	1.91/1	7.33/1
k -means++	0.2 ± 0.4	0.3 ± 0.7	0.2 ± 0.4	0.6 ± 0.7
1000 execs in s/relQ	7.47/1.85	13.46/2.07	26.66/1.67	51.85/2.41
k -means++ 2 ×	0 ± 0	0 ± 0	0 ± 0	0 ± 0
1000 execs in s/relQ	14.85/1	26.82/1	53.17/1	103.47/1

Other parameters fixed at $k = 5$ $d = 2$ $breakgap = 1$ $p = 0.1$

Let us make a comparison to the results of other investigators in the realm of well-clusterability, in particular presented in [1, 2, 7–9, 15, 20].

If the data is well clusterable according to criteria of Perturbation Robustness, or ϵ -Separatedness, or (c, ϵ) -Approximation-Stability or α -Centre Stability or $(1 + \alpha)$ Weak Deletion Stability or Perfect Separation, one can reconstruct the well-clustered structure using appropriate algorithm. But only in case of Perfect Separation or Nice Separation, you can decide that you have found the structure, if you have

found it. Note that you have no warranty that you will find Nice Separation if it is there. But for none of these ways of understanding well-clusterability one is able to decide (neither a priori nor a posteriori) that the data is not well-clusterable if the well-clustered structure was not found (unless by brute force).

The only exception constitutes formally the method of Multi-modality Detection, as proposed in [1], which tells us a priori that the data is or is not well-clusterable. However, as we have demonstrated, by the example in Fig. 1 in

Table 14 Dependence of the number of errors on *breakgap*

<i>breakgap</i> =	1	2	4	8
Hartigan–Wong 1 ×	737.8 ± 36.7	704.8 ± 62.6	712.5 ± 62.9	711.6 ± 41.7
1000 execs in s/relQ	0.49/14832.58	0.49/1527.1	0.47/874.07	0.47/221.09
Lloyd 1 ×	744.5 ± 29.1	724.8 ± 52.9	726.7 ± 53.6	731.8 ± 28.8
1000 execs in s/relQ	0.48/15121.58	0.47/1830.33	0.46/923.38	0.46/246.31
Forgy 1 ×	743.1 ± 30	717.1 ± 60.7	714.1 ± 48.2	729.5 ± 42
1000 execs in s/relQ	0.48/15136.66	0.48/1764	0.46/938.35	0.46/239.11
MacQueen 1 ×	736.7 ± 21.8	721.8 ± 58.6	721.6 ± 57.6	727.1 ± 31.7
1000 execs in s/relQ	0.48/14539.53	0.48/1824.64	0.48/946.43	0.45/235.61
Hartigan–Wong 10 ×	50.5 ± 19	36.2 ± 17.4	39.1 ± 20	44.4 ± 17.4
1000 execs in s/relQ	1.56/99.45	1.5/22.34	1.44/6.98	1.43/3.32
Lloyd 10 ×	51.8 ± 14.8	43.8 ± 25	45.4 ± 28.5	49.9 ± 19.2
1000 execs in s/relQ	1.41/101.54	1.39/27.1	1.32/7.92	1.34/3.57
Forgy 10 ×	55.2 ± 15.7	42.9 ± 20.1	43.3 ± 26.1	47.7 ± 20.2
1000 execs in s/relQ	1.41/108.27	1.36/26.29	1.32/7.58	1.33/3.49
MacQueen 10 ×	48.1 ± 19.2	39.6 ± 21.8	40.5 ± 25.7	41.8 ± 19.2
1000 execs in s/relQ	1.39/94.4	1.33/24.45	1.29/7.16	1.29/3.26
Hartigan–Wong 20 ×	1.9 ± 1.6	2.4 ± 1.9	1.8 ± 2.3	2.1 ± 2.1
1000 execs in s/relQ	1.96/4.68	1.85/2.4	1.8/1.27	1.8/1.11
Lloyd 20 ×	2.5 ± 2.3	2.9 ± 2.9	2.2 ± 3	2.4 ± 2.6
1000 execs in s/relQ	1.7/5.76	1.65/2.7	1.59/1.35	1.61/1.12
Forgy 20 ×	2.9 ± 2.6	2.5 ± 2	2.5 ± 3.1	2.5 ± 3.2
1000 execs in s/relQ	1.68/6.6	1.63/2.46	1.61/1.39	1.61/1.13
MacQueen 20 ×	2.9 ± 2	2.7 ± 2.7	2.4 ± 2.7	3.5 ± 3
1000 execs in s/relQ	1.65/6.61	1.56/2.58	1.53/1.36	1.52/1.18
cmeans Fuzzy	48 ± 90	49.6 ± 85.4	132 ± 140.6	146.3 ± 104.9
1000 execs in s/relQ	1.38/95.98	1.35/27.62	1.31/23.67	1.34/8.83
ufcl Fuzzy	784.5 ± 13.2	775.7 ± 18.6	800.6 ± 21.1	832 ± 34.2
1000 execs in s/relQ	2.17/27243.69	2.04/3671.76	2/1786.46	2/363.3
single link	0 ± 0	0 ± 0	0 ± 0	0 ± 0
1000 execs in s/relQ	0.48/1	0.44/1	0.43/1	0.44/1
<i>k</i> -means++	0.3 ± 0.7	0.9 ± 0.9	4.7 ± 2.4	12.1 ± 4.3
1000 execs in s/relQ	10.38/1.56	9.58/1.63	9.39/1.84	9.39/1.67
<i>k</i> -means++ 2 ×	0 ± 0	0 ± 0	0 ± 0	0.2 ± 0.4
1000 execs in s/relQ	20.91/1	18.95/1	18.75/1	18.72/1.01

Other parameters fixed at $k = 5$ $d = 2$ $M = 15$, $m = 9$ $p = 0.1$

“Failure of Multi-modality Detection for Detection of Gaps Between Data” section, data can be easily found that can fool this method, so that it discovers well-clusterability in case when there is none.

Under the definitions of well-clusterability presented in this paper, one gets a completely new situation. It is guaranteed that if the well-clusterable structure is there, it will be detected with high probability. One can check a posteriori that the structure found is the well-clusterable structure if it

is so, with 100% certainty. Furthermore, if the (*k*-means++) algorithm did not find a well-clusterable structure then with high probability it is not there in the data.

The paper contains a couple of other, minor contributions. The concept of cluster cores has been introduced such that if a seed of *k*-means once hits each core then there is guarantee that none of the cluster centres will ever leave the cluster. It has been shown that the number of reruns of *k*-means++ is small when a desired probability of success in finding the

Table 15 Dependence of the number of errors on p

$p=$	0.1	0.2	0.3	0.4
Hartigan–Wong 1 ×	709.6 ± 40.7	739.7 ± 37.2	727.8 ± 40	727.3 ± 30.9
1000 execs in s/relQ	0.51/9331.26	0.53/6214.92	0.52/14411.23	0.49/11570.51
Lloyd 1 ×	730.6 ± 38.3	754.2 ± 37.9	725.1 ± 35.1	745.7 ± 37.2
1000 execs in s/relQ	0.49/10181.63	0.51/6489.79	0.51/14022.92	0.47/11998.16
Forgy 1 ×	723.8 ± 38.2	749.9 ± 46.6	725.1 ± 37	736 ± 30.3
1000 execs in s/relQ	0.49/10529.17	0.51/6483.41	0.51/15049.05	0.47/12702.78
MacQueen 1 ×	733.6 ± 32.4	745.6 ± 40.3	728 ± 37	737.4 ± 31.9
1000 execs in s/relQ	0.49/10804.58	0.5/6239.37	0.5/15511.64	0.48/12367.06
Hartigan–Wong 10 ×	38.3 ± 23.1	55.4 ± 21.6	47.5 ± 24.9	48.1 ± 12.9
1000 execs in s/relQ	1.57/81.12	1.72/131.9	1.65/133.48	1.6/158.6
Lloyd 10 ×	48.1 ± 25.7	56.7 ± 25.1	48.2 ± 23.9	55.3 ± 21.6
1000 execs in s/relQ	1.49/105.41	1.55/133.51	1.51/134.32	1.47/177.88
Forgy 10 ×	47.9 ± 19.8	57.8 ± 21.5	46.7 ± 23.7	52.8 ± 20.9
1000 execs in s/relQ	1.5/104.24	1.53/136.46	1.5/130.72	1.45/170.55
MacQueen 10 ×	46.6 ± 22.5	59.9 ± 19.6	44.5 ± 23.2	51.3 ± 20.2
1000 execs in s/relQ	1.49/102.86	1.52/142.14	1.49/124.33	1.45/164.45
Hartigan–Wong 20 ×	1.6 ± 3.1	2.5 ± 2.1	2.2 ± 2.9	1.9 ± 2.1
1000 execs in s/relQ	2.02/4.26	2.09/6.75	2.05/6.96	2/6.88
Lloyd 20 ×	2.7 ± 3.5	4.1 ± 2.6	1.6 ± 2.5	3.3 ± 2.3
1000 execs in s/relQ	1.78/6.54	1.83/10.53	1.78/5.15	1.72/11.47
Forgy 20 ×	2.6 ± 3.3	3.9 ± 2.8	1.7 ± 2.2	3.9 ± 2.8
1000 execs in s/relQ	1.76/6.32	1.84/10.02	1.79/5.56	1.73/13.36
MacQueen 20 ×	3.2 ± 2.9	4.8 ± 3.4	2.8 ± 3.8	2.4 ± 2.6
1000 execs in s/relQ	1.71/7.68	1.83/11.97	1.73/8.47	1.68/8.64
cmeans Fuzzy	43.1 ± 85.3	21.7 ± 37.8	41.7 ± 93.2	37.3 ± 80
1000 execs in s/relQ	1.4/89.96	1.48/54.06	1.44/116.27	1.4/116.36
ufcl Fuzzy	786.1 ± 26.5	796.3 ± 18	773.8 ± 22.8	772.9 ± 24.4
1000 execs in s/relQ	2.21/22096.93	2.39/11322.8	2.3/25817.26	2.25/31905.38
single link	0 ± 0	0 ± 0	0 ± 0	0 ± 0
1000 execs in s/relQ	0.49/1	0.54/1	0.51/1	0.5/1
k -means++	0.6 ± 0.7	0 ± 0	0.4 ± 0.5	0.1 ± 0.3
1000 execs in s/relQ	10.55/2.4	11.75/1	11.07/2.6	10.98/1.31
k -means++ 2 ×	0 ± 0	0 ± 0	0 ± 0	0 ± 0
1000 execs in s/relQ	21.23/1	23.47/1	22.01/1	21.63/1

Other parameters fixed at $k = 5$ $d = 2$ $M = 15$, $m = 9$ $breakgap = 1$

well-clusterability is being targeted. Numerical examples show that several orders of magnitude smaller gaps between clusters, compared to [20], are required in order to state that the data is well clusterable, and still the probability of detecting the well-clusterable structure is much higher (even close to one in a single k -means++ run).

The procedure elaborated for constructing a well-clusterable data set, ensuring that the k -means cost function absolute minimum is reached for a predefined data partition may find applications in some testing procedures of clustering algorithms.

Of course a number of research questions with respect to the topic of this paper remain open. First is the issue of

constructing tight (or at least tighter) bounds for estimation of required gaps between clusters. Second is an investigation how the violations of these minimum values influence the capability of k -means algorithms to detect either the absolute minimum of their cost function or achieving a partition that is intuitively considered by humans as “good clustering”.

Funding Funding was provided by Polska Akademia Nauk.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

- Ackerman M, Adolfsson A, Brownstein N. An effective and efficient approach for clusterability evaluation. CoRR [arXiv:abs/1602.06687](https://arxiv.org/abs/1602.06687); 2016.
- Ackerman M, Ben-David S. Clusterability: a theoretical study. In: van Dyk D, Welling M, editors. Proceedings of the twelfth international conference on artificial intelligence and statistics, volume 5 of proceedings of machine learning research. Hilton Clearwater Beach Resort, Clearwater Beach, Florida, USA, 16–18 Apr 2009. PMLR; 2009, p. 1–8.
- Ackerman M, Ben-David S, Loker D, Sabato S. Clustering oligarchies. In: Proceedings of the sixteenth international conference on Artificial Intelligence and Statistics, AISTATS 2013, Scottsdale, AZ, USA, April 29–May 1 2013; 2013, p. 66–74.
- Ackerman M, Dasgupta S. Incremental clustering: the case for extra clusters. In: Advances in neural information processing systems 27: annual conference on neural information processing systems 2014, December 8–13 2014, Montreal, Quebec, Canada; 2014, p. 307–315.
- Arthur D, Vassilvitskii S. k -means++: the advantages of careful seeding. In Bansal N, Pruhs K, Stein C, editors. Proceedings of the eighteenth annual ACM-SIAM symposium on discrete algorithms, SODA 2007, New Orleans, Louisiana, USA, 7–9 Jan 2007. SIAM; 2007, p. 1027–1035.
- Awasthi P, Blum A, Sheffet O. Stability yields a PTAS for k -median and k -means clustering. In: Proceedings of the 2010 IEEE 51st annual symposium on foundations of computer science, FOCS '10, Washington, DC, USA. IEEE Computer Society; 2010, p. 309–318.
- Awasthi P, Blum A, Sheffet O. Center-based clustering under perturbation stability. Inf Process Lett. 2012;112(1–2):49–54.
- Balcan M-F, Blum A, Gupta A. Approximate clustering without the approximation. In: Proceedings of the twentieth annual ACM-SIAM symposium on discrete algorithms, SODA 2009, New York, NY, USA, January 4–6, 2009; 2009, p. 1068–1077.
- Balcan M-F, Blum A, Vempala S. A discriminative framework for clustering via similarity functions. In: Proceedings of the fortieth annual ACM symposium on theory of computing, STOC '08, New York, NY, USA. ACM; 2008, p. 671–680.
- Ben-David S. Computational feasibility of clustering under clusterability assumptions. [arXiv:abs/1501.00437](https://arxiv.org/abs/1501.00437); 2015.
- Ben-David S, Haghtala N. Clustering in the presence of background noise. In: Proceedings of the 31st international conference on machine learning, Beijing, China, 2014. JMLR: W&CP volume 32; 2014.
- Bilu Y, Linial N. Are stable instances easy? Comb Probab Comput. 2012;21(5):643–60.
- Chakraborty S, Paul D, Das S, Xu J. Entropy regularized power k -means clustering. [arXiv:2001.03452](https://arxiv.org/abs/2001.03452); 2020.
- Cohen-Addad V. On the local structure of stable clustering instances. <https://arxiv.org/pdf/1701.08423.pdf>; 2017.
- Epter S, Krishnamoorthy M, Zaki M. Clusterability detection and initial seed selection in large datasets; 1999.
- Hartigan JA, Hartigan PM. The dip test of unimodality. Ann Stat. 1985;13(1):70–84.
- Kłopotek R, Kłopotek M. On the discrepancy between Kleinberg's clustering axioms and k -means clustering algorithm behavior. [arXiv:abs/1702.04577](https://arxiv.org/abs/1702.04577); 2017.
- Kumar A, Kannan R. Clustering with spectral norm and the k -means algorithm. In: 51st annual IEEE symposium on foundations of Computer Science, FOCS 2010, Las Vegas, Nevada, USA; 2010, p. 299–308.
- Kushagra S, Samadi S, Ben-David S. Finding meaningful cluster structure amidst background noise. In Algorithmic learning theory (ALT), volume 9925 of Lecture Notes in Computer Science. Cham: Springer; 2016, p. 2016.
- Ostrovsky R, Rabani Y, Schulman LJ, Swamy C. The effectiveness of Lloyd-type methods for the k -means problem. J. ACM. 2013;59(6):28:1–2.
- Silverman BW. Using kernel density estimates to investigate multimodality. J R Stat Soci Ser B (Methodol.). 1981;43(1):97–9.
- Tang C, Monteleoni C. On Lloyd's algorithm: new theoretical insights for clustering in practice. In: Proceedings of the 19th international conference on Artificial Intelligence and Statistics (AISTATS) 2016, Cadiz, Spain. JMLR: W&CP volume 41; 2016.
- von Luxburg U. A tutorial on spectral clustering. <https://arxiv.org/pdf/0711.0189.pdf>; 2007.
- Wierzchoń ST, Kłopotek MA. Algorithms of Cluster Analysis. Monograph Series 3. Institute of Computer Science Publishing House. Warszawa; 2015.
- Xu J, Lange K. Power k -means clustering. In Chaudhuri K, Salakhutdinov R, editors. Proceedings of the 36th international conference on machine learning, volume 97 of proceedings of machine learning research, Long Beach, California, USA, 09–15 Jun 2019. PMLR, p. 6921–6931.
- Zhang B, Hsu M, Dayal U. K -harmonic means—a data clustering algorithm. Technical Report HPL-1999-124, Software Technology Laboratory, HP Laboratories Palo Alto; 1999.

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