Chapter 3
Linear, Deterministic, and Order-Invariant Initialization Methods for the K-Means Clustering Algorithm

M. Emre Celebi and Hassan A. Kingravi

Abstract Over the past five decades, k-means has become the clustering algorithm of choice in many application domains primarily due to its simplicity, time/space efficiency, and invariance to the ordering of the data points. Unfortunately, the algorithm's sensitivity to the initial selection of the cluster centers remains to be its most serious drawback. Numerous initialization methods have been proposed to address this drawback. Many of these methods, however, have time complexity superlinear in the number of data points, which makes them impractical for large data sets. On the other hand, linear methods are often random and/or sensitive to the ordering of the data points. These methods are generally unreliable in that the quality of their results is unpredictable. Therefore, it is common practice to perform multiple runs of such methods and take the output of the run that produces the best results. Such a practice, however, greatly increases the computational requirements of the otherwise highly efficient k-means algorithm. In this chapter, we investigate the empirical performance of six linear, deterministic (non-random), and order-invariant k-means initialization methods on a large and diverse collection of data sets from the UCI Machine Learning Repository. The results demonstrate that two relatively unknown hierarchical initialization methods due to Su and Dy outperform the remaining four methods with respect to two objective effectiveness criteria. In addition, a recent method due to Erişoğlu et al. performs surprisingly poorly.

Keywords Data mining • Unsupervised learning • Clustering • K-means • Cluster center initialization • Maximin

---

M.E. Celebi (✉)
Department of Computer Science, Louisiana State University, Shreveport, LA, USA
e-mail: ecelebi@lsus.edu

H.A. Kingravi
School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA, USA
e-mail: kingravi@gatech.edu

M.E. Celebi (ed.), *Partitional Clustering Algorithms*,
DOI 10.1007/978-3-319-09259-1_3

3.1 Introduction

Clustering, the unsupervised classification of patterns into groups, is one of the most important tasks in exploratory data analysis [59]. Primary goals of clustering include gaining insight into, classifying, and compressing data. Clustering has a long and rich history in a variety of scientific disciplines including anthropology, biology, medicine, psychology, statistics, mathematics, engineering, and computer science. As a result, numerous clustering algorithms have been proposed since the early 1950s [58].

Clustering algorithms can be broadly classified into two groups: hierarchical and partitional [59]. Hierarchical algorithms recursively find nested clusters either in a top-down (divisive) or bottom-up (agglomerative) fashion. In contrast, partitional algorithms find all the clusters simultaneously as a partition of the data and do not impose a hierarchical structure. Most hierarchical algorithms have time complexity quadratic or higher in the number of data points [111] and therefore are not suitable for large data sets, whereas partitional algorithms often have lower complexity.

Given a data set $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\} \subset \mathbb{R}^D$, i.e., N points (vectors) each with D attributes (components), hard partitional algorithms divide $\mathcal{X}$ into K exhaustive and mutually exclusive clusters $\mathcal{C} = \{\mathcal{C}_1, \mathcal{C}_2, \dots, \mathcal{C}_K\}$, $\quad \bigcup_{i=1}^{K} \mathcal{C}_i = \mathcal{X}$, $\quad \mathcal{C}_i \cap \mathcal{C}_j = \varnothing$ for $1 \leq i \neq j \leq K$. These algorithms usually generate clusters by optimizing a criterion function [48]. The most intuitive and frequently used criterion function is the Sum of Squared Error (SSE) given by

$$\mathrm{SSE} = \sum_{i=1}^{K} \sum_{\mathbf{x}_j \in \mathcal{C}_i} \left\| \mathbf{x}_j - \mathbf{c}_i \right\|_2^2, \tag{3.1}$$

where

$$\mathbf{c}_i = \frac{1}{|\mathcal{C}_i|} \sum_{\mathbf{x}_j \in \mathcal{C}_i} \mathbf{x}_j \tag{3.2}$$

and

$$\left\| \mathbf{x}_j \right\|_2 = \left(\sum_{d=1}^{D} x_{jd}^2 \right)^{1/2} \tag{3.3}$$

denote the centroid of cluster $\mathcal{C}_i$ (with cardinality $|\mathcal{C}_i|$) and the Euclidean (ℓ_2) norm of vector $\mathbf{x}_j = (x_{j1}, x_{j2}, \dots, x_{jD})$, respectively.

The number of ways in which a set of N objects can be partitioned into K non-empty groups is given by Stirling numbers of the second kind

$$\mathcal{S}(N, K) = \frac{1}{K!} \sum_{i=0}^{K} (-1)^{K-i} \binom{K}{i} i^N, \tag{3.4}$$

which can be approximated by $K^N/K!$ It can be seen that a complete enumeration of all possible clusterings to determine the global minimum of (3.1) is clearly computationally prohibitive except for very small data sets. In fact, this non-convex optimization problem is proven to be NP-hard even for $K = 2$ [4, 30] or $D = 2$ [79, 106]. Consequently, various heuristics have been developed to provide approximate solutions to this problem [102]. Most of the early approaches [12, 39, 53, 61, 77, 78, 98, 100] were simple procedures based on the alternating minimization algorithm [28]. In contrast, recent approaches are predominantly based on various metaheuristics [29, 94] that are capable of avoiding bad local minima at the expense of significantly increased computational requirements. These include heuristics based on simulated annealing [70], evolution strategies [10], tabu search [3], genetic algorithms [85], variable neighborhood search [51], memetic algorithms [90], scatter search [89], ant colony optimization [50], differential evolution [91], and particle swarm optimization [91]. Among all these heuristics, Lloyd's algorithm [77], often referred to as the (batch) k-means algorithm, is the simplest and most commonly used one. This algorithm starts with K arbitrary centers, typically chosen uniformly at random from the data points. Each point is assigned to the nearest center and then each center is recalculated as the mean of all points assigned to it. These two steps are repeated until a predefined termination criterion is met. K-means can be expressed in algorithmic notation as follows:

1. Choose the initial set of centers $\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_K$ arbitrarily.
2. Assign point $\mathbf{x}_j$ ($j \in \{1, 2, \ldots, N\}$) to the nearest center with respect to ℓ_2 distance, that is

$$\mathbf{x}_j \in \mathcal{C}_{\hat{\imath}} \iff \hat{\imath} = \underset{i \in \{1,2,\ldots,K\}}{\arg\min} \left\| \mathbf{x}_j - \mathbf{c}_i \right\|_2^2 .$$

3. Recalculate center $\mathbf{c}_i$ ($i \in \{1, 2, \ldots, K\}$) as the centroid of $\mathcal{C}_i$, that is

$$\mathbf{c}_i = \frac{1}{|\mathcal{C}_i|} \sum_{\mathbf{x}_j \in \mathcal{C}_i} \mathbf{x}_j .$$

4. Repeat steps 2 and 3 until convergence.

K-means is undoubtedly the most widely used partitional clustering algorithm [15, 17, 41, 48, 58, 59, 86, 110, 111]. Its popularity can be attributed to several reasons. First, it is conceptually simple and easy to implement. Virtually every data mining software includes an implementation of it. Second, it is versatile, i.e., almost every aspect of the algorithm (initialization, distance function, termination criterion, etc.) can be modified. This is evidenced by hundreds of publications over the last fifty years that extend k-means in a variety of ways. Third, it has a time complexity that is linear in N, D, and K (in general, $D \ll N$ and $K \ll N$). For this reason, it can be used to initialize more expensive clustering algorithms such as expectation maximization [82], fuzzy c-means [16, p. 35], DBSCAN [31], spectral clustering [27, 108], ant colony clustering[84], and particle swarm clustering [105]. Furthermore, numerous sequential [34,35,49,63,66,72,92] and parallel [5, 14, 26, 46, 57, 71, 74, 109] acceleration techniques are available in the literature. Fourth, it has a storage complexity that is linear in N, D, and K. In addition, there exist disk-based variants that do not require all points to be stored in memory [20, 38, 62, 88]. Fifth, it is guaranteed to converge [97] at a quadratic rate [18]. Finally, it is invariant to data ordering, i.e., random shufflings of the data points.

On the other hand, k-means has several significant disadvantages. First, it requires the number of clusters, K, to be specified in advance. The value of this parameter can be determined automatically by means of various internal/relative cluster validity measures [6, 9, 107]. Second, it can only detect compact, hyperspherical clusters that are well separated. This can be alleviated by using a more general distance function such as the Mahalanobis distance, which permits the detection of hyperellipsoidal clusters [80, 81]. Third, due its utilization of the squared Euclidean distance, it is sensitive to noise and outlier points since even a few such points can significantly influence the means of their respective clusters. This can be addressed by outlier pruning [112] or by using a more robust distance function such as the city-block (ℓ_1) distance [37, 60, 99]. Fourth, due to its gradient descent nature, it often converges to a local minimum of the criterion function [97]. For the same reason, it is highly sensitive to the selection of the initial centers [25]. Adverse effects of improper initialization include empty clusters, slower convergence, and a higher chance of getting stuck in bad local minima [23]. Fortunately, except for the first two, these drawbacks can be remedied by using an adaptive initialization method (IM).

A large number of IMs have been proposed in the literature [23,25,32,54,93]. Unfortunately, many of these have time complexity superlinear in N [1, 2, 8, 21, 53, 65, 68, 73, 75, 95], which makes them impractical for large data sets (note that k-means itself has linear time complexity). In contrast, linear IMs are often random and/or order-sensitive [7, 11, 19, 39, 61, 78, 100, 104], which renders their results unreliable. In this study, we investigate the empirical performance of six linear, deterministic (non-random), and order-invariant k-means IMs on a large and diverse collection of data sets from the UCI Machine Learning Repository.

The rest of the chapter is organized as follows. Section 3.2 presents an overview of linear, deterministic, and order-invariant k-means IMs. Section 3.3 describes the experimental setup. Section 3.4 presents and discusses the experimental results. Finally, Sect. 3.5 gives the conclusions.

3.2 Linear, Deterministic, and Order-Invariant K-Means Initialization Methods

In this study, we focus on IMs that have time complexity linear in N. This is because k-means itself has linear complexity, which is perhaps the most important reason for its popularity. Therefore, an IM for k-means should not diminish this advantage of the algorithm. Accordingly, the following six linear, deterministic, and order-invariant IMs are investigated.

The maximin (MM) method [47] chooses the first center $\mathbf{c}_1$ arbitrarily from the data points and the remaining $(K-1)$ centers are chosen successively as follows. In iteration i ($i \in \{2, 3, \dots, K\}$), the ith center $\mathbf{c}_i$ is chosen to be the point with the greatest minimum ℓ_2 distance to the previously selected $(i-1)$ centers, i.e., $\mathbf{c}_1, \mathbf{c}_2, \dots, \mathbf{c}_{i-1}$. This method can be expressed in algorithmic notation as follows:

1. Choose the first center $\mathbf{c}_1$ arbitrarily from the data points.
2. Choose the next center $\mathbf{c}_i$ ($i \in \{2, 3, \dots, K\}$) as the point $\mathbf{x}_{\hat{\jmath}}$ that satisfies

$$\hat{\jmath} = \underset{j \in \{1,2,\dots,N\}}{\arg\max} \left(\min_{k \in \{1,2,\dots,i-1\}} \left\| \mathbf{x}_j - \mathbf{c}_k \right\|_2^2 \right).$$

3. Repeat step 2 $(K-1)$ times.

Despite the fact that it was originally developed as a 2-approximation to the K-center clustering problem,[1] MM is commonly used as a k-means initializer.[2] In this study, the first center is chosen to be the centroid of $\mathcal{X}$ given by

$$\bar{\mathbf{x}} = \frac{1}{N} \sum_{j=1}^{N} \mathbf{x}_j . \tag{3.5}$$

Note that $\mathbf{c}_1 = \bar{\mathbf{x}}$ gives the optimal SSE when $K = 1$.

Katsavounidis et al.'s method (KK) [67] is identical to MM with the exception that the first center is chosen to be the point with the greatest ℓ_2 norm,[3] that is, the point $\mathbf{x}_{\hat{\jmath}}$ that satisfies

$$\hat{\jmath} = \underset{j \in \{1,2,\dots,N\}}{\arg\max} \left\| \mathbf{x}_j \right\|_2^2 . \tag{3.6}$$

The PCA-Part (PP) method [101] uses a divisive hierarchical approach based on Principal Component Analysis (PCA) [64]. Starting from an initial cluster that contains the entire data set $\mathcal{X}$, the method successively selects the cluster with the greatest SSE and divides it into two subclusters using a hyperplane that passes

[1] Given a set of N points in a metric space, the goal of K-center clustering is to find K representative points (centers) such that the maximum distance of a point to a center is minimized [52, p. 63]. A polynomial-time algorithm is said to be a δ-*approximation* algorithm for a minimization problem if for every instance of the problem it delivers a solution whose cost is at most δ times the cost of the optimal solution (δ is often referred to as the "approximation ratio" or "approximation factor") [55, p. xv].

[2] Interestingly, several authors including Thorndike [103], Casey and Nagy [22], Batchelor and Wilkins [13], Kennard and Stone [69], and Tou and Gonzalez [104, pp. 92–94] had proposed similar (or even identical) methods decades earlier. Gonzalez [47], however, was the one to prove the theoretical properties of the method.

[3] This choice was motivated by a vector quantization application.

through the cluster centroid and is orthogonal to the principal eigenvector of the cluster covariance matrix. This iterative cluster selection and splitting procedure is repeated $(K - 1)$ times. The final centers are then given by the centroids of the resulting K subclusters. This method can be expressed in algorithmic notation as follows:

1. Let $\mathcal{C}_i$ be the cluster with the greatest SSE and $\mathbf{c}_i$ be the centroid of this cluster. In the first iteration, $\mathcal{C}_1 = \mathcal{X}$ and $\mathbf{c}_1 = \bar{\mathbf{x}}$.
2. Let p be the projection of $\mathbf{c}_i$ on the principal eigenvector $\mathbf{v}_i$ of $\mathcal{C}_i$, i.e., $p = \mathbf{c}_i \cdot \mathbf{v}_i$, where '$\cdot$' denotes the dot product.
3. Divide $\mathcal{C}_i$ into two subclusters $\mathcal{C}_{i_1}$ and $\mathcal{C}_{i_2}$ according to the following rule: For any $\mathbf{x}_j \in \mathcal{C}_i$, if $\mathbf{x}_j \cdot \mathbf{v}_i \leq p$, then assign $\mathbf{x}_j$ to $\mathcal{C}_{i_1}$; otherwise, assign it to $\mathcal{C}_{i_2}$.
4. Repeat steps 1–3 $(K - 1)$ times.

The Var-Part (VP) method [101] is an approximation to PP, where, in each iteration, the covariance matrix of the cluster to be split is assumed to be diagonal. In this case, the splitting hyperplane is orthogonal to the coordinate axis with the greatest variance. In other words, the only difference between VP and PP is the choice of the projection axis.

Figure 3.1 [24] illustrates VP on a toy data set with four natural clusters [96][68, p. 100]. In iteration 1, the initial cluster that contains the entire data set is split into two subclusters along the Y axis using a line (i.e., a one-dimensional hyperplane) passing through the mean point (92.026667). Between the resulting two clusters, the one above the line has a greater SSE. In iteration 2, this cluster is thus split along the X axis at the mean point (66.975000). In the final iteration, the cluster with the greatest SSE, i.e., the bottom cluster, is split along the X axis at the mean point (41.057143). In Fig. 3.1d, the centroids of the final four clusters are denoted by stars.

The maxisum (MS) method [36] is a recent modification of MM. It can be expressed in algorithmic notation as follows:

1. Determine the attribute with the greatest absolute *coefficient of variation* (ratio of the standard deviation to the mean), that is, the attribute $\mathbf{x}_{.d_1}$ that satisfies

$$d_1 = \underset{d \in \{1,2,\ldots,D\}}{\arg\max} \left| \frac{s_d}{m_d} \right|,$$

where

$$m_d = \frac{1}{N} \sum_{j=1}^{N} x_{jd}$$

and

$$s_d^2 = \frac{1}{N-1} \sum_{j=1}^{N} (x_{jd} - m_d)^2$$

denote the mean and variance of the dth attribute $\mathbf{x}_{.d}$, respectively.

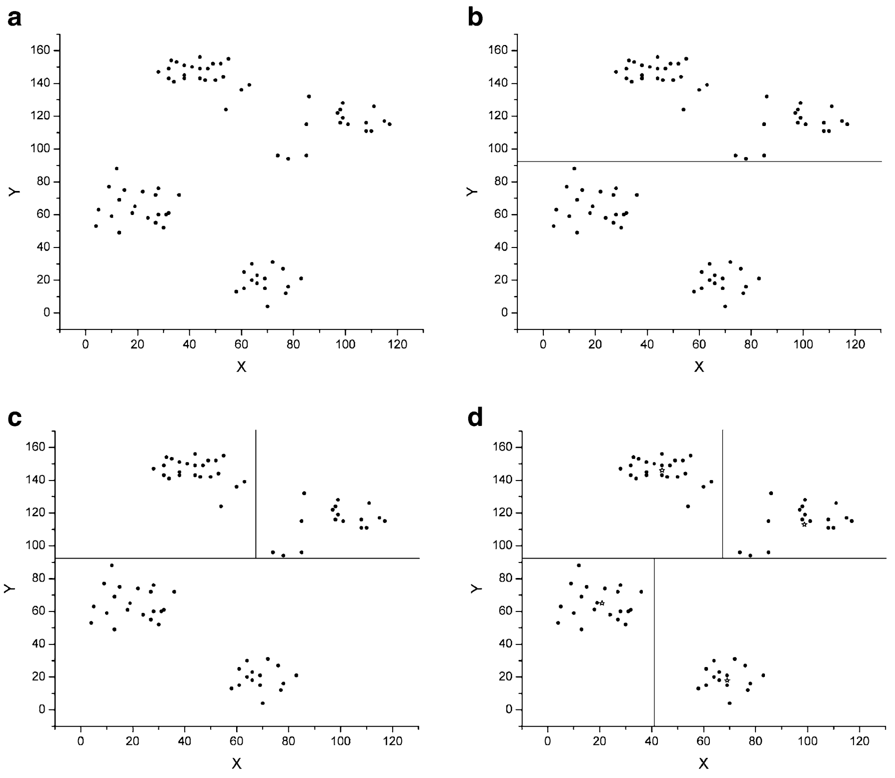

Fig. 3.1 Illustration of Var-Part on the Ruspini data set. (**a**) Input data set. (**b**) Iteration 1. (**c**) Iteration 2. (**d**) Iteration 3

2. Determine the attribute with the least *Pearson product-moment correlation* with $\mathbf{x}_{.d_1}$, that is, the attribute $\mathbf{x}_{.d_2}$ that satisfies

$$d_2 = \underset{d \in \{1,2,\ldots,D\}}{\arg\min} \frac{\sum_{j=1}^{N}(x_{jd_1} - m_{d_1})(x_{jd} - m_d)}{\sqrt{\sum_{j=1}^{N}(x_{jd_1} - m_{d_1})^2}\sqrt{\sum_{j=1}^{N}(x_{jd} - m_d)^2}}. \tag{3.7}$$

Note that since we calculated the mean and standard deviation of each attribute in step 1, the following expression can be used in place of (3.7) to save computational time:

$$d_2 = \underset{d \in \{1,2,\ldots,D\}}{\arg\min} \sum_{j=1}^{N} \left(\frac{x_{jd_1} - m_{d_1}}{s_{d_1}}\right)\left(\frac{x_{jd} - m_d}{s_d}\right). \tag{3.8}$$

3. Let $\mathcal{Y} = \{\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_N\} \subset \mathbb{R}^2$ be the projection of $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\} \subset \mathbb{R}^D$ onto the two-dimensional subspace determined in steps 1 and 2. In other words, $\mathbf{y}_j = \left(x_{jd_1}, x_{jd_2}\right)$ for $j \in \{1, 2, \dots, N\}$.
4. Choose the first center $\mathbf{c}_1$ as the point farthest from the centroid $\bar{\mathbf{y}}$ of $\mathcal{Y}$ with respect to ℓ_2 distance, that is, the point $\mathbf{y}_{\hat{j}}$ that satisfies

$$\hat{j} = \underset{j \in \{1,2,\dots,N\}}{\arg\max} \left\| \mathbf{y}_j - \bar{\mathbf{y}} \right\|_2^2 .$$

5. Choose the next center $\mathbf{c}_i$ ($i \in \{2, 3, \dots, K\}$) as the point with the greatest cumulative ℓ_2 distance from the previously selected $(i-1)$ centers, that is, the point $\mathbf{y}_{\hat{j}}$ that satisfies

$$\hat{j} = \underset{j \in \{1,2,\dots,N\}}{\arg\max} \sum_{k=1}^{i-1} \left\| \mathbf{y}_j - \mathbf{c}_k \right\|_2 .$$

6. Repeat step 5 $(K-1)$ times.

Note that steps 1 and 2 above provide rough approximations to the first two PCs and that steps 4 and 5 are performed in the two-dimensional subspace spanned by the attributes determined in steps 1 and 2.

Clearly, `MS` is a derivative of `MM`. Differences between the two methods are as follows:

- `MM` chooses the first center arbitrarily from the data points, whereas `MS` chooses it to be the point farthest from the mean of the projected data set.
- `MM` chooses the remaining $(K-1)$ centers iteratively based on their minimum distance from the previously selected centers, whereas `MS` uses a cumulative distance criterion. Note that while the selection criterion used in `MM` provides an approximation guarantee of factor 2 for the K-center clustering problem (see footnote 1 on page 83), it is unclear whether or not `MS` offers any approximation guarantees.
- `MM` performs the distance calculations in the original D-dimensional space, whereas `MS` works in a two-dimensional subspace. A serious drawback of the projection operation employed in `MS` is that the method disregards all attributes but two and therefore is likely to be effective only for data sets in which the variability is mostly on two dimensions. Unfortunately, the motivation behind this particular projection scheme is not given by Erişoğlu et al.

Interestingly, `MS` also bears a striking resemblance to a method proposed by DeSarbo et al. [33] almost three decades earlier. The latter method differs from the former in two ways. First, it works in the original D-dimensional space. Second, it chooses the first two centers as the pair of points with the greatest ℓ_2 distance. Unfortunately, the determination of the first two centers in this method leads to a time complexity quadratic in N. Therefore, this method was not included in the experiments. More recently, Glasbey et al. [43] mentioned a very similar method within the context of color palette design.

We also experimented with a modified version of the `MS` method (`MS+`), which is identical to `MS` with the exception that there is no projection involved. In other words, `MS+` operates in the original D-dimensional space.

For a comprehensive overview of these methods and others, the reader is referred to a recent article by Celebi et al. [25]. It should be noted that, in this study, we do not attempt to compare a mix of deterministic and random IMs. Instead, we focus on deterministic methods for two main reasons. First, these methods are generally computationally more efficient as they need to be executed only once. In contrast, random methods are inherently unreliable in that the quality of their results is unpredictable and thus it is common practice to perform multiple runs of such methods and take the output of the run[4] that produces the best objective function value. Second, several studies [24, 25, 101] demonstrated that despite the fact that they are executed only once, some deterministic methods are highly competitive with well-known and effective random methods such as Bradley and Fayyad's method [19] and k-means++ [7].

3.3 Experimental Setup

3.3.1 Data Set Descriptions

The experiments were performed on 24 commonly used data sets from the UCI Machine Learning Repository [40]. Table 3.1 gives the data set descriptions. For each data set, the number of clusters (K) was set equal to the number of classes (K'), as commonly seen in the related literature [2, 7, 21, 24, 25, 42, 54, 65, 87, 95, 101].

3.3.2 Attribute Normalization

Normalization is a common preprocessing step in clustering that is necessary to prevent attributes with large variability from dominating the distance calculations and also to avoid numerical instabilities in the computations. Two commonly used normalization schemes are linear scaling to unit range (min-max normalization) and linear scaling to unit variance (z-score normalization). Several studies revealed that the former scheme is preferable to the latter since the latter is likely to eliminate valuable between-cluster variation [44, 45, 83]. As a result, we used the min-max normalization scheme to map the attributes of each data set to the [0, 1] interval.

[4]Each 'run' of a random IM involves the execution of the IM itself followed by that of the clustering algorithm, e.g., k-means.

Table 3.1 Data set descriptions (N: # points, D: # attributes, K': # classes)

ID	Data Set	N	D	K'
01	Breast cancer Wisconsin (original)	683	9	2
02	Breast tissue	106	9	6
03	Ecoli	336	7	8
04	Steel plates faults	1,941	27	7
05	Glass identification	214	9	6
06	Heart disease (Cleveland)	297	13	5
07	Ionosphere	351	34	2
08	Iris (Bezdek)	150	4	3
09	ISOLET	7,797	617	26
10	Landsat satellite (Statlog)	6,435	36	6
11	Letter recognition	20,000	16	26
12	Multiple features (Fourier)	2,000	76	10
13	Libras movement	360	90	15
14	Optical digits	5,620	64	10
15	Page blocks classification	5,473	10	5
16	Pen digits	10,992	16	10
17	Person activity	164,860	3	11
18	Image segmentation	2,310	19	7
19	Shuttle (Statlog)	58,000	9	7
20	Spambase	4,601	57	2
21	Vertebral column	310	6	3
22	Wall-following robot navigation	5,456	24	4
23	Wine	178	13	3
24	Yeast	1,484	8	10

3.3.3 Performance Criteria

The performance of the IMs was quantified using two effectiveness (quality) and one efficiency (speed) criteria:

- **Initial SSE (IS)**: This is the SSE value calculated after the initialization phase, before the clustering phase. It gives us a measure of the effectiveness of an IM by itself.
- **Final SSE (FS)**: This is the SSE value calculated after the clustering phase. It gives us a measure of the effectiveness of an IM when its output is refined by k-means. Note that this is the objective function of the k-means algorithm, i.e., (3.1).
- **Number of Iterations (NI)**: This is the number of iterations that k-means requires until reaching convergence when initialized by a particular IM. It is an efficiency measure independent of programming language, implementation

style, compiler, and CPU architecture. Note that we do not report CPU time measurements since on most data sets that we tested each of the six IMs completed within a few milliseconds (gcc v4.4.5, Intel Core i7-3960X 3.30GHz).

The convergence of k-means was controlled by the disjunction of two criteria: the number of iterations reaches a maximum of 100 or the relative improvement in SSE between two consecutive iterations drops below a threshold [76], i.e., $(\mathrm{SSE}_{i-1} - \mathrm{SSE}_i) / \mathrm{SSE}_i \leq \epsilon$, where SSE_i denotes the SSE value at the end of the ith ($i \in \{2, \ldots, 100\}$) iteration. The convergence threshold was set to $\epsilon = 10^{-6}$.

3.4 Experimental Results and Discussion

Tables 3.2, 3.3, and 3.4 give the performance measurements for each method (the best values are underlined). Since the number of iterations fall within $[0, 100]$, we can directly obtain descriptive statistics such as minimum, maximum, mean, and median for this criterion over the 24 data sets. In contrast, initial/final SSE values are unnormalized and therefore incomparable across different data sets. In order to circumvent this problem, for each data set, we calculated the percent SSE of each method relative to the worst (greatest) SSE. For example, it can be seen from Table 3.2 that on the Breast Cancer Wisconsin data set the initial SSE of `MM` is 498, whereas the worst initial SSE on the same data set is 596 and thus the ratio of the former to the latter is 0.836. This simply means that on this data set `MM` obtains $100(1 - 0.836) \approx 16\,\%$ better initial SSE than the worst method, `KK`. Table 3.5 gives the summary statistics for the normalized initial/final SSE's obtained in this manner and those for the number of iterations. As usual, *min* (minimum) and *max* (maximum) represent the *best* and *worst* case performance, respectively. *Mean* represents the *average* case performance, whereas median quantifies the *typical* performance of a method without regard to outliers. For example, with respect to the initial SSE criterion, `PP` performs, on the *average*, about $100 - 21.46 \approx 79\,\%$ better than the worst method.

For convenient visualization, Fig. 3.2 shows the box plots that depict the five-number summaries (minimum, 25th percentile, median, 75th percentile, and maximum) for the normalized initial/final SSE's calculated in the aforementioned manner and the five-number summary for the number of iterations. Here, the bottom and top end of the whiskers of a box represent the *minimum* and *maximum*, respectively, whereas the bottom and top of the box itself are the 25th percentile (*Q1*) and 75th percentile (*Q3*), respectively. The line that passes through the box is the 50th percentile (*Q2*), i.e., the *median*, while the small square inside the box denotes the *mean*.

With respect to effectiveness, the following observations can be made:

- `VP` and `PP` performed very similarly with respect to both initial and final SSE.

Table 3.2 Initial SSE comparison of the initialization methods

ID	MM	KK	VP	PP	MS	MS+
01	498	596	247	$\underline{240}$	478	596
02	19	18	$\underline{8}$	$\underline{8}$	50	21
03	48	76	20	$\underline{19}$	104	68
04	2,817	3,788	$\underline{1,203}$	1,262	5,260	4,627
05	45	117	21	$\underline{20}$	83	132
06	409	557	$\underline{249}$	250	773	559
07	827	1,791	632	$\underline{629}$	3,244	3,390
08	18	23	$\underline{8}$	$\underline{8}$	42	42
09	221,163	298,478	145,444	$\underline{124,958}$	368,510	318,162
10	4,816	7,780	$\underline{2,050}$	2,116	7,685	11,079
11	5,632	7,583	3,456	$\underline{3,101}$	12,810	14,336
12	4,485	7,821	3,354	$\underline{3,266}$	7,129	8,369
13	1,023	1,114	628	$\underline{592}$	1,906	1,454
14	25,291	36,691	17,476	$\underline{15,714}$	43,169	42,213
15	635	2,343	300	$\underline{230}$	1,328	7,868
16	12,315	16,159	5,947	$\underline{5,920}$	17,914	16,104
17	5,940	7,196	$\underline{1,269}$	1,468	42,475	50,878
18	1,085	1,617	472	$\underline{416}$	3,071	1,830
19	1,818	14,824	316	$\underline{309}$	26,778	28,223
20	$\underline{772}$	13,155	782	783	5,101	13,155
21	37	103	23	$\underline{20}$	83	103
22	11,004	21,141	8,517	$\underline{7,805}$	19,986	20,122
23	87	185	$\underline{51}$	53	153	212
24	115	261	77	$\underline{63}$	209	658

- On 23 (out of 24) data sets, VP and PP obtained the two best initial SSE's. Therefore, in applications where an approximate clustering of the data set is desired, these hierarchical methods should be used.
- On 23 data sets, either MS or MS+ obtained the worst initial SSE. In fact, on one data set (#19, Shuttle), these methods gave respectively 86.7 and 91.3 times worse initial SSE than the best method, PP.
- On 20 data sets, VP and PP obtained the two best final SSE's. Since final SSE is the objective function of k-means, from an optimization point of view, these two methods are the best IMs.

Table 3.3 Final SSE comparison of the initialization methods

ID	MM	KK	VP	PP	MS	MS+
01	$\underline{239}$	$\underline{239}$	$\underline{239}$	$\underline{239}$	$\underline{239}$	$\underline{239}$
02	$\underline{7}$	$\underline{7}$	$\underline{7}$	$\underline{7}$	11	10
03	19	20	$\underline{17}$	18	40	20
04	1, 331	1, 329	$\underline{1,167}$	1, 168	1, 801	1, 376
05	23	23	$\underline{19}$	$\underline{19}$	31	22
06	249	249	248	$\underline{243}$	276	253
07	826	$\underline{629}$	$\underline{629}$	$\underline{629}$	$\underline{629}$	$\underline{629}$
08	$\underline{7}$	$\underline{7}$	$\underline{7}$	$\underline{7}$	$\underline{7}$	$\underline{7}$
09	135, 818	123, 607	118, 495	$\underline{118,386}$	174, 326	121, 912
10	$\underline{1,742}$	$\underline{1,742}$	$\underline{1,742}$	$\underline{1,742}$	$\underline{1,742}$	$\underline{1,742}$
11	2, 749	2, 783	$\underline{2,735}$	2, 745	4, 520	3, 262
12	3, 316	3, 284	$\underline{3,137}$	3, 214	3, 518	3, 257
13	502	502	502	$\underline{486}$	783	530
14	14, 679	14, 649	$\underline{14,581}$	14, 807	21, 855	$\underline{14,581}$
15	230	295	227	$\underline{215}$	230	310
16	5, 049	$\underline{4,930}$	$\underline{4,930}$	5, 004	7, 530	5, 017
17	1, 195	1, 195	1, 182	$\underline{1,177}$	1, 226	1, 192
18	433	443	410	$\underline{405}$	745	446
19	726	658	$\underline{235}$	274	728	496
20	$\underline{765}$	$\underline{765}$	778	778	778	$\underline{765}$
21	23	23	$\underline{19}$	$\underline{19}$	23	23
22	$\underline{7,772}$	$\underline{7,772}$	7, 774	7, 774	$\underline{7,772}$	$\underline{7,772}$
23	63	$\underline{49}$	$\underline{49}$	$\underline{49}$	$\underline{49}$	$\underline{49}$
24	61	61	69	$\underline{59}$	60	63

- On 16 data sets, `MS` obtained the worst final SSE. In fact, on one data set (#19, Shuttle), `MS` gave 3.1 times worse final SSE than the best method, `VP`.
- A comparison between Fig. 3.2a, b reveals that there is significantly less variation among the IMs with respect to final SSE compared to initial SSE. In other words, the performance of the IMs is more homogeneous with respect to final SSE. This was expected because, being a local optimization procedure, k-means can take two disparate initial configurations to similar (or, in some cases, even identical) local minima. Nevertheless, as Tables 3.2 and 3.3 show, `VP` and `PP` consistently performed well, whereas `MS`/`MS+` generally performed poorly.

With respect to computational efficiency, the following observations can be made:

- An *average* (or *typical*) run of `KK` lead to the fastest k-means convergence.
- An *average* (or *typical*) run of `PP` lead to the second fastest k-means convergence.
- An *average* run of `MS` lead to the slowest k-means convergence.
- A *typical* run of `MM` lead to the slowest k-means convergence.

Table 3.4 Number of iterations comparison of the initialization methods

ID	MM	KK	VP	PP	MS	MS+
01	8	7	4	4	7	7
02	7	6	6	7	9	4
03	14	12	17	7	4	10
04	25	16	11	42	12	12
05	6	5	6	5	7	6
06	12	10	3	4	11	16
07	3	6	3	3	7	6
08	6	5	4	4	12	19
09	32	36	82	45	34	81
10	53	17	28	27	24	33
11	72	63	100	83	91	65
12	37	32	14	25	31	29
13	13	7	17	11	18	16
14	36	24	16	22	29	17
15	27	18	25	15	30	12
16	19	17	13	17	22	29
17	31	31	100	63	91	53
18	31	9	10	18	16	22
19	22	8	30	16	14	9
20	5	5	9	10	11	5
21	11	10	10	9	8	10
22	24	14	20	8	20	19
23	9	7	5	7	7	8
24	73	43	33	21	71	49

In summary, our experiments showed that VP and PP performed very similarly with respective to both effectiveness criteria and they outperformed the remaining four methods by a large margin. The former method has a time complexity of $\mathcal{O}(ND)$, whereas the latter one has a complexity of $\mathcal{O}(ND^2)$ when implemented using the power method [56]. Therefore, on high dimensional data sets, the former method might be preferable. On the other hand, on low dimensional data sets, the latter method might be preferable as it often leads to faster k-means convergence. The main disadvantage of these two methods is that they are more complicated to implement due to their hierarchical formulation. As for the remaining four methods, when compared to MM, KK was significantly worse in terms of initial SSE, slightly better in terms of final SSE, and significantly better in terms of number of iterations. Interestingly, despite its similarities with MM, the most recent method that we examined, i.e., MS, often gave the worst results. It was also demonstrated that by eliminating the two-dimensional projection step, the performance of MS can be substantially improved with respect to final SSE. This, however, comes at the expense of a performance degradation with respect to initial SSE. Consequently, in either of its forms, the MS method rediscovered recently by

Table 3.5 Summary statistics for Tables 3.2, 3.3, and 3.4

Criterion	Statistic	MM	KK	VP	PP	MS	MS+
IS	Min	5.87	14.14	1.12	1.10	16.88	42.00
	Q1	29.24	52.74	15.64	14.35	76.19	87.15
	Median	41.95	72.04	20.78	19.26	94.71	100.00
	Q3	53.57	89.42	33.07	32.69	100.00	100.00
	Max	83.56	100.00	41.44	40.27	100.00	100.00
	Mean	40.28	69.03	22.55	21.46	83.16	90.53
FS	Min	47.50	50.00	32.28	37.64	74.19	50.00
	Q1	67.11	66.25	63.87	62.85	100.00	69.03
	Median	89.31	83.09	74.69	72.75	100.00	84.34
	Q3	99.99	97.90	98.21	93.68	100.00	99.15
	Max	100.00	100.00	100.00	100.00	100.00	100.00
	Mean	83.21	81.56	76.23	75.77	96.46	82.68
NI	Min	3.00	5.00	3.00	3.00	4.00	4.00
	Q1	8.50	7.00	6.00	7.00	8.50	8.50
	Median	20.50	11.00	13.50	13.00	15.00	16.00
	Q3	31.50	21.00	26.50	23.50	29.50	29.00
	Max	73.00	63.00	100.00	83.00	91.00	81.00
	Mean	24.00	17.00	23.58	19.71	24.42	22.38

Erişoğlu et al. does not appear to outperform the classical `MM` method or the more recent hierarchical methods `VP` and `PP`. This is not surprising given that `MS` can easily choose two nearby points as centers provided that they each have a large cumulative distance to all other centers [43].

3.5 Conclusions

In this chapter we examined six linear, deterministic, and order-invariant methods used for the initialization of the k-means clustering algorithm. These included the popular maximin method and three of its variants and two relatively unknown divisive hierarchical methods. Experiments on a large and diverse collection of real-world data sets from the UCI Machine Learning Repository demonstrated that the hierarchical methods outperform the remaining four methods with respect to two objective effectiveness criteria. These hierarchical methods can be used to initialize k-means effectively, particularly in time-critical applications that involve large data sets. Alternatively, they can be used as approximate clustering algorithms without additional k-means refinement. Our experiments also revealed that the most recent variant of the maximin method performs surprisingly poorly.

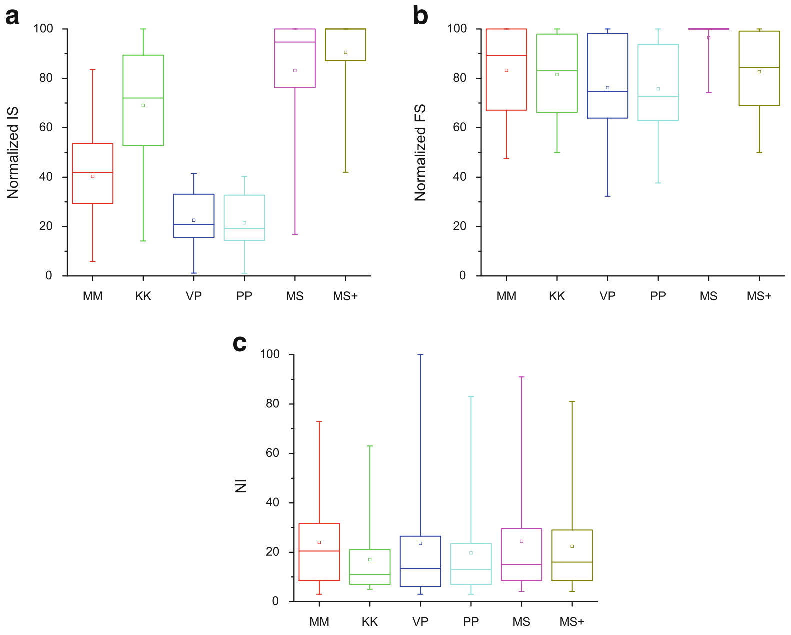

Fig. 3.2 Box plots for the performance criteria. (**a**) Normalized initial SSE. (**b**) Normalized final SSE. (**c**) Number of iterations

Acknowledgements This work was supported by a grant from the US National Science Foundation (1117457).

References

1. Al-Daoud M (2005) A new algorithm for cluster initialization. In: Proceedings of the 2nd world enformatika conference, pp 74–76
2. Al Hasan M, Chaoji V, Salem S, Zaki M (2009) Robust partitional clustering by outlier and density insensitive seeding. Pattern Recognit Lett 30(11):994–1002
3. Al-Sultan KS (1995) A Tabu search approach for the minimum sum-of-squares clustering problem. Pattern Recognit 28(9):1443–1451
4. Aloise D, Deshpande A, Hansen P, Popat P (2009) NP-hardness of Euclidean sum-of-squares clustering. Mach Learn 75(2):245–248
5. An F, Mattausch HJ (2013) K-means clustering algorithm for multimedia applications with flexible HW/SW co-design. J Syst Archit 59(3):155–164
6. Arbelaitz O, Gurrutxaga I, Muguerza J, Pérez JM, Perona I (2013) An extensive comparative study of cluster validity indices. Pattern Recognit 46(1):243–256
7. Arthur D, Vassilvitskii S (2007) K-means++: the advantages of careful seeding. In: Proceedings of the 18th annual ACM-SIAM symposium on discrete algorithms, pp 1027–1035
8. Astrahan MM (1970) Speech analysis by clustering, or the hyperphoneme method. Technical Report AIM-124, Stanford University

9. Baarsch J, Celebi ME (2012) Investigation of internal validity measures for K-means clustering. In: Proceedings of the 2012 IAENG international conference on data mining and applications, pp 471–476
10. Babu GP, Murty MN (1994) Clustering with evolution strategies. Pattern Recognit 27(2): 321–329
11. Ball GH, Hall DJ (1967) A clustering technique for summarizing multivariate data. Behav Sci 12(2):153–155
12. Banfield CF, Bassill LC (1977) A transfer algorithm for non-hierarchical classification. Appl Stat 26(2):206–210
13. Batchelor BG, Wilkins BR (1969) Method for location of clusters of patterns to initialise a learning machine. Electron Lett 5(20):481–483
14. Bekkerman R, Bilenko M, Langford J (eds) (2012) Scaling up machine learning: parallel and distributed approaches. Cambridge University Press, Cambridge
15. Berkhin P (2006) A survey of clustering data mining techniques. In: Kogan J, Nicholas C, Teboulle M (eds) Grouping multidimensional data: recent advances in clustering. Springer, Berlin, pp 25–71
16. Bezdek JC, Keller J, Krisnapuram R, Pal NR (1999) Fuzzy models and algorithms for pattern recognition and image processing. Kluwer, Dordecht
17. Bock HH (2007) Clustering methods: a history of K-means algorithms. In: Brito P, Cucumel G, Bertrand P, de Carvalho F (eds) Selected contributions in data analysis and classification. Springer, Berlin, pp 161–172
18. Bottou L, Bengio Y (1995) Convergence properties of the K-means algorithms. In: Tesauro G, Touretzky DS, Leen TK (eds) Advances in neural information processing systems, vol 7. MIT Press, Cambridge, pp 585–592
19. Bradley PS, Fayyad U (1998) Refining initial points for K-means clustering. In: Proceedings of the 15th international conference on machine learning, pp 91–99
20. Bradley PS, Fayyad U, Reina C (1998) Scaling clustering algorithms to large databases. In: Proceedings of the 4th international conference on knowledge discovery and data mining, pp 9–15
21. Cao F, Liang J, Jiang G (2009) An initialization method for the K-means algorithm using neighborhood model. Comput Math Appl 58(3):474–483
22. Casey RG, Nagy G (1968) An autonomous reading machine. IEEE Trans Comput 17(5): 492–503
23. Celebi ME (2011) Improving the performance of K-means for color quantization. Image Vis Comput 29(4):260–271
24. Celebi ME, Kingravi HA (2012) Deterministic initialization of the K-means algorithm using hierarchical clustering. Intern J Pattern Recognit Artif Intell 26(7):1250018
25. Celebi ME, Kingravi HA, Vela PA (2013) A Comparative study of efficient initialization methods for the K-means clustering algorithm. Expert Syst Appl 40(1):200–210
26. Chen TW, Chien SY (2010) Bandwidth adaptive hardware architecture of K-means clustering for video analysis. IEEE Trans VLSI Syst 18(6):957–966
27. Chen WY, Song Y, Bai H, Lin CJ, Chang EY (2011) Parallel spectral clustering in distributed systems. IEEE Trans Pattern Anal Mach Intell 33(3):568–586
28. Csiszar I, Tusnady G (1984) Information geometry and alternating minimization procedures. Stat Decis Suppl Issue (1):205–237
29. Das S, Abraham A, Konar A (eds) (2009) Metaheuristic clustering. Springer, Berlin
30. Dasgupta S (2008) The hardness of K-means clustering. Technical Report CS2008-0916, University of California, San Diego
31. Dash M, Liu H, Xu X (2001) '$1 + 1 > 2$': merging distance and density based clustering. In: Proceedings of the 7th international conference on database systems for advanced applications, pp 32–39
32. de Amorim RC, Komisarczuk P (2012) On Initializations for the Minkowski weighted K-means. In: Proceedings of the 11th international symposium on intelligent data analysis, pp 45–55

33. DeSarbo WS, Carroll JD, Clark LA, Green PE (1984) Synthesized clustering: a method for amalgamating alternative clustering bases with differential weighting of variables. Psychometrika 49(1):57–78
34. Drake J, Hamerly G (2012) Accelerated K-means with adaptive distance bounds. In: Proceedings of the 5th NIPS workshop on optimization for machine learning
35. Elkan C (2003) Using the triangle inequality to accelerate K-means. In: Proceedings of the 20th international conference on machine learning, pp 147–153
36. Erişoğlu et al. (2011) Pattern Recognit Lett 32(14):1701–1705
37. Estivill-Castro V, Yang J (2004) Fast and robust general purpose clustering algorithms. Data Min Knowl Discov 8(2):127–150
38. Farnstrom F, Lewis J, Elkan C (2000) Scalability for clustering algorithms revisited. SIGKDD Explor 2(1):51–57
39. Forgy E (1965) Cluster analysis of multivariate data: efficiency vs. interpretability of classification. Biometrics 21:768
40. Frank A, Asuncion A (2014) UCI machine learning repository. School of Information and Computer Sciences, University of California, Irvine. http://archive.ics.uci.edu/ml
41. Ghosh J, Liu A (2009) K-Means. In: Wu X, Kumar V (eds) The top ten algorithms in data mining. Chapman and Hall/CRC, London, pp 21–35
42. Gingles C, Celebi ME (2014) Histogram-based method for effective initialization of the K-means clustering algorithm. In: Proceedings of the 27th international Florida artificial intelligence research society conference, pp 333–338
43. Glasbey C, van der Heijden G, Toh VFK, Gray A (2006) Colour displays for categorical images. Color Res Appl 32(4):304–309
44. Gnanadesikan R, Kettenring JR (1995) Weighting and selection of variables for cluster analysis. J Classif 12(1):113–136
45. Gnanadesikan R, Kettenring JR, Maloor S (2007) Better alternatives to current methods of scaling and weighting data for cluster analysis. J Stat Plan Inference 137(11):3483–3496
46. Gokhale M, Frigo J, McCabe, K., Theiler J, Wolinski C, Lavenier D (2003) Experience with a hybrid processor: K-means clustering. J Supercomput 26(2):131–148
47. Gonzalez T (1985) Clustering to minimize the maximum intercluster distance. Theor Comput Sci 38(2–3):293–306
48. Hall LO (2012) Objective function-based clustering. WIREs Data Min Knowl Discov 2(4):326–339
49. Hamerly G (2010) Making K-means even faster. In: Proceedings of the 2010 SIAM international conference on data mining, pp 130–140
50. Handl J, Knowles J, Dorigo M (2005) Ant-based clustering and topographic mapping. Artif Life 12(1):35–61
51. Hansen P, Mladenovic N (2001) J-means: a new local search heuristic for minimum sum of squares clustering. Pattern Recognit 34(2):405–413
52. Har-Peled S (2011) Geometric approximation algorithms. American Mathematical Society, Providence
53. Hartigan JA, Wong MA (1979) Algorithm AS 136: a K-means clustering algorithm. J R Stat Soc C 28(1):100–108
54. He J, Lan M, Tan CL, Sung SY, Low HB (2004) Initialization of cluster refinement algorithms: a review and comparative study. In: Proceedings of the 2004 IEEE international joint conference on neural networks, pp 297–302
55. Hochbaum DS (ed) (1997) Approximation algorithms for NP-hard problems. PWS Publishing Company, Boston
56. Hotelling H (1936) Simplified calculation of principal components. Psychometrika 1(1): 27–35
57. Hwang WJ, Hsu CC, Li HY, Weng SK, Yu TY (2010) High speed C-means clustering in reconfigurable hardware. Microprocess Microsyst 34(6):237–246
58. Jain AK (2010) Data clustering: 50 years beyond K-means. Pattern Recognit Lett 31(8): 651–666

59. Jain AK, Murty MN, Flynn PJ (1999) Data clustering: a review. ACM Comput Surv 31(3):264–323
60. Jajuga K (1987) A clustering method based on the L_1-norm. Comput Stat Data Anal 5(4): 357–371
61. Jancey RC (1966) Multidimensional group analysis. Aust J Bot 14(1):127–130
62. Jin R, Goswami A, Agrawal G (2006) Fast and exact out-of-core and distributed K-means clustering. Knowl Inf Syst 10(1):17–40
63. Jin X, Kim S, Han J, Cao L, Yin Z (2011) A general framework for efficient clustering of large datasets based on activity detection. Stat Anal Data Min 4(1):11–29
64. Jolliffe IT (2002) Principal component analysis, 2nd edn. Springer, Berlin
65. Kang P, Cho S (2009) K-means clustering seeds initialization based on centrality, sparsity, and isotropy. In: Proceedings of the 10th international conference on intelligent data engineering and automated learning, pp 109–117
66. Kanungo T, Mount D, Netanyahu N, Piatko C, Silverman R, Wu A (2002) An efficient K-means clustering algorithm: analysis and implementation. IEEE Trans Pattern Anal Mach Intell 24(7):881–892
67. Katsavounidis I, Kuo CCJ., Zhang Z (1994) A new initialization technique for generalized Lloyd iteration. IEEE Signal Process Lett 1(10):144–146
68. Kaufman L, Rousseeuw P (1990) Finding groups in data: an introduction to cluster analysis. Wiley-Interscience, London
69. Kennard RW, Stone LA (1969) Computer aided design of experiments. Technometrics 11(1):137–148
70. Klein RW, Dubes RC (1989) Experiments in projection and clustering by simulated annealing. Pattern Recognit 22(2):213–220
71. Kohlhoff KJ, Pande VS, Altman RB (2013) K-means for parallel architectures using all-prefix-sum sorting and updating steps. IEEE Trans Parallel Distrib Syst 24(8):1602–1612
72. Lai JZC., Huang TJ, Liaw YC (2009) A fast K-means clustering algorithm using cluster center displacement. Pattern Recognit 42(11):2551–2556
73. Lance GN, Williams WT (1967) A general theory of classificatory sorting strategies - II. Clustering systems. Comput J 10(3):271–277
74. Li Y, Zhao K, Chu X, Liu J (2013) Speeding up K-means algorithm by GPUs. J Comput Syst Sci 79(2):216–229
75. Likas A, Vlassis N, Verbeek J (2003) The global K-means clustering algorithm. Pattern Recognit 36(2):451–461
76. Linde Y, Buzo A, Gray R (1980) An algorithm for vector quantizer design. IEEE Trans Commun 28(1):84–95
77. Lloyd S (1982) Least squares quantization in PCM. IEEE Trans Inf Theory 28(2):129–136
78. MacQueen J (1967) Some methods for classification and analysis of multivariate observations. In: Proceedings of the 5th Berkeley symposium on mathematical statistics and probability, pp 281–297
79. Mahajan M, Nimbhorkar P, Varadarajan K (2012) The planar k-means problem is NP-hard. Theor Comput Sci 442, 13–21
80. Mao J, Jain AK (1996) A self-organizing network for hyperellipsoidal clustering (HEC). IEEE Trans Neural Netw 7(1):16–29
81. Melnykov I, Melnykov V (2014) On K-means algorithm with the use of Mahalanobis distances. Stat Probab Lett 84, 88–95
82. Melnykov V (2013) Challenges in model-based clustering. Wiley Interdiscip Rev Comput Stat 5(2):135–148
83. Milligan G, Cooper MC (1988) A study of standardization of variables in cluster analysis. J Classif 5(2):181–204
84. Monmarché N, Slimane M, Venturini G (1999) On improving clustering in numerical databases with artificial ants. In: Proceedings of the 5th European conference on advances in artificial life, pp 626–635
85. Murthy CA, Chowdhury N (1996) In search of optimal clusters using genetic algorithms. Pattern Recognit Lett 17(8):825–832

86. Omran MGH, Engelbrecht AP, Salman A (2007) An overview of clustering methods. Intell Data Anal 11(6):583–605
87. Onoda T, Sakai M, Yamada S (2012) Careful seeding method based on independent components analysis for K-means clustering. J Emerg Technol Web Intell 4(1):51–59
88. Ordonez C, Omiecinski E (2004) Efficient disk-based K-means clustering for relational databases. IEEE Trans Knowl Data Eng 16(8):909–921
89. Pacheco JA (2005) A scatter search approach for the minimum sum-of-squares clustering problem. Comput Oper Res 32(5):1325–1335
90. Pacheco JA, Valencia O (2003) Design of hybrids for the minimum sum-of-squares clustering problem. Comput Stat Data Anal 43(2):235–248
91. Paterlini S, Krink T (2006) Differential evolution and particle swarm optimization in partitional clustering. Comput Stat Data Anal 50(5):1220–1247
92. Pelleg D, Moore A (1999) Accelerating exact K-means algorithms with geometric reasoning. In: Proceedings of the 5th ACM SIGKDD international conference on knowledge discovery and data mining, pp 277–281
93. Pena JM, Lozano JA, Larranaga P (1999) An empirical comparison of four initialization methods for the K-means algorithm. Pattern Recognit Lett 20(10):1027–1040
94. Rayward-Smith VJ (2005) Metaheuristics for clustering in KDD. In: Proceedings of the IEEE congress on evolutionary computation, vol. 3, pp 2380–2387
95. Redmond SJ, Heneghan C (2007) A method for initialising the K-means clustering algorithm using kd-trees. Pattern Recognit Lett 28(8):965–973
96. Ruspini EH (1970) Numerical methods for fuzzy clustering. Inf Sci 2(3):319–350
97. Selim SZ, Ismail MA (1984) K-means-type algorithms: a generalized convergence theorem and characterization of local optimality. IEEE Trans Pattern Anal Mach Intell 6(1):81–87
98. Sparks DN (1973) Euclidean cluster analysis. Appl Stat 22(1):126–130
99. Späth H (1976) L1 cluster analysis. Computing 16(4):379–387
100. Späth H (1977) Computational experiences with the exchange method: applied to four commonly used partitioning cluster analysis criteria. Eur J Oper Res 1(1):23–31
101. Su T, Dy JG (2007) In search of deterministic methods for initializing K-means and Gaussian mixture clustering. Intell Data Anal 11(4):319–338
102. Tarsitano A (2003) A computational study of several relocation methods for K-means algorithms. Pattern Recognit 36(12):2955–2966
103. Thorndike RL (1953) Who belongs in the family? Psychometrika 18(4):267–276
104. Tou JT, Gonzales RC (1974) Pattern recognition principles. Addison-Wesley, Reading
105. van der Merwe, D.W., Engelbrecht AP (2003) Data clustering using particle swarm optimization. In: Proceedings of the 2003 IEEE congress on evolutionary computation, pp 215–220
106. Vattani A (2011) K-means requires exponentially many iterations even in the plane. Discrete Comput Geom 45(4):596–616
107. Vendramin L, Campello RJG.B., Hruschka ER (2010) Relative clustering validity criteria: a comparative overview. Stat Anal Data Min 3(4):209–235
108. von Luxburg, U (2007) A tutorial on spectral clustering. Stat Comput 17(4):395–416
109. Wu R, Zhang B, Hsu M (2009) Clustering billions of data points using GPUs. In: Proceedings of the 6th ACM conference on computing Frontiers, pp 1–6
110. Xiao Y, Yu J (2012) Partitive clustering (K-means family). WIREs Data Min Knowl Discov 2(3):209–225
111. Xu R, Wunsch D (2005) Survey of clustering algorithms. IEEE Trans Neural Netw 16(3): 645–678
112. Zhang JS, Leung YW (2003) Robust clustering by pruning outliers. IEEE Trans Syst Man Cybern B 33(6):983–999