Expert Systems with Applications 40 (2013) 200–210

Contents lists available at SciVerse ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

A comparative study of efficient initialization methods for the k-means clustering algorithm

M. Emre Celebi [a,*], Hassan A. Kingravi [b], Patricio A. Vela [b]

[a] Department of Computer Science, Louisiana State University, Shreveport, LA, USA
[b] School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA, USA

ARTICLE INFO

Keywords:
Partitional clustering
Sum of squared error criterion
k-means
Cluster center initialization

ABSTRACT

K-means is undoubtedly the most widely used partitional clustering algorithm. Unfortunately, due to its gradient descent nature, this algorithm is highly sensitive to the initial placement of the cluster centers. Numerous initialization methods have been proposed to address this problem. In this paper, we first present an overview of these methods with an emphasis on their computational efficiency. We then compare eight commonly used linear time complexity initialization methods on a large and diverse collection of data sets using various performance criteria. Finally, we analyze the experimental results using non-parametric statistical tests and provide recommendations for practitioners. We demonstrate that popular initialization methods often perform poorly and that there are in fact strong alternatives to these methods.

1. Introduction

Clustering, the unsupervised classification of patterns into groups, is one of the most important tasks in exploratory data analysis (Jain, Murty, & Flynn, 1999). Primary goals of clustering include gaining insight into data (detecting anomalies, identifying salient features, etc.), classifying data, 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, a plethora of clustering algorithms have been proposed since the early 1950s (Jain, 2010).

Clustering algorithms can be broadly classified into two groups: hierarchical and partitional (Jain, 2010). 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 quadratic or higher complexity in the number of data points (Jain et al., 1999) and therefore are not suitable for large data sets, whereas partitional algorithms often have lower complexity.

Given a data set $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$ in $\mathbb{R}^D$, i.e. N points (vectors) each with D attributes (components), hard partitional algorithms divide $\mathcal{X}$ into K exhaustive and mutually exclusive clusters $\mathcal{P} = \{P_1, P_2, \ldots, P_K\}$, $\bigcup_{i=1}^{K} P_i = \mathcal{X}$, $P_i \cap P_j = \emptyset$ for $1 \leqslant i \neq j \leqslant K$. These algorithms usually generate clusters by optimizing a criterion function. The most intuitive and frequently used criterion function is the Sum of Squared Error (SSE) given by:

$$\mathrm{SSE} = \sum_{i=1}^{K} \sum_{\mathbf{x}_j \in P_i} \|\mathbf{x}_j - \mathbf{c}_i\|_2^2 \tag{1}$$

where $\|\cdot\|_2$ denotes the Euclidean ($\mathcal{L}_2$) norm and $\mathbf{c}_i = 1/|P_i| \sum_{\mathbf{x}_j \in P_i} \mathbf{x}_j$ is the centroid of cluster P_i whose cardinality is $|P_i|$. The optimization of (1) is often referred to as the minimum SSE clustering (MSSC) problem.

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:

$$\mathcal{S}(N,K) = \frac{1}{K!} \sum_{i=0}^{K} (-1)^{K-i} \binom{K}{i} i^N \tag{2}$$

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 (1) is clearly computationally prohibitive except for very small data sets (Kaufman & Rousseeuw, 1990). In fact, this non-convex optimization problem is proven to be NP-hard even for $K = 2$ (Aloise, Deshpande, Hansen, & Popat, 2009) or $D = 2$ (Mahajan, Nimbhorkar, & Varadarajan, 2012). Consequently, various heuristics have been developed to provide approximate solutions to this problem (Tarsitano, 2003). Among these heuristics, Lloyd's algorithm (Lloyd, 1982), 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

* Corresponding author.
E-mail addresses: ecelebi@lsus.edu (M.E. Celebi), kingravi@gatech.edu (H.A. Kingravi), pvela@gatech.edu (P.A. Vela).

0957-4174/$ - see front matter © 2012 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.eswa.2012.07.021

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.

The k-means algorithm is undoubtedly the most widely used partitional clustering algorithm (Jain et al., 1999; Jain, 2010). 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 various 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 (Bradley & Fayyad, 1998), DBSCAN (Dash, Liu, & Xu, 2001), and spectral clustering (Chen, Song, Bai, Lin, & Chang, 2011). Furthermore, numerous sequential (Kanungo et al., 2002; Hamerly, 2010) and parallel (Chen & Chien, 2010) 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 (Ordonez & Omiecinski, 2004). Fifth, it is guaranteed to converge (Selim & Ismail, 1984) at a quadratic rate (Bottou & Bengio, 1995). 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 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 (Mao & Jain, 1996). Second, 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 addressed by outlier pruning (Zhang & Leung, 2003) or using a more robust distance function such as City-block ($\mathcal{L}_1$) distance. Third, due to its gradient descent nature, it often converges to a local minimum of the criterion function (Selim & Ismail, 1984). For the same reason, it is highly sensitive to the selection of the initial centers. Adverse effects of improper initialization include empty clusters, slower convergence, and a higher chance of getting stuck in bad local minima (Celebi, 2011). Fortunately, all of these drawbacks except for the first one can be remedied by using an adaptive initialization method (IM).

In this study, we investigate some of the most popular IMs developed for the k-means algorithm. Our motivation is threefold. First, a large number of IMs have been proposed in the literature and thus a systematic study that reviews and compares these methods is desirable. Second, these IMs can be used to initialize other partitional clustering algorithms such as fuzzy c-means and its variants and expectation maximization. Third, most of these IMs can be used independently of k-means as standalone clustering algorithms.

This study differs from earlier studies of a similar nature (Pena, Lozano, & Larranaga, 1999; He, Lan, Tan, Sung, & Low, 2004) in several respects: (i) a more comprehensive overview of the existing IMs is provided, (ii) the experiments involve a larger set of methods and a significantly more diverse collection of data sets, (iii) in addition to clustering effectiveness, computational efficiency is used as a performance criterion, and (iv) the experimental results are analyzed more thoroughly using non-parametric statistical tests.

The rest of the paper is organized as follows. Section 2 presents a survey of k-means IMs. Section 3 describes the experimental setup. Section 4 presents the experimental results, while Section 5 gives the conclusions.

2. Initialization methods for k-means

In this section, we briefly review some of the commonly used IMs with an emphasis on their time complexity (with respect to N). In each complexity class, methods are presented in chronologically ascending order.

2.1. Linear time-complexity initialization methods

Forgy's method (Forgy, 1965) assigns each point to one of the K clusters uniformly at random. The centers are then given by the centroids of these initial clusters. This method has no theoretical basis, as such random clusters have no internal homogeneity (Anderberg, 1973).

Jancey's method (Jancey, 1966) assigns to each center a synthetic point randomly generated within the data space. Unless the data set fills the space, some of these centers may be quite distant from any of the points (Anderberg, 1973), which might lead to the formation of empty clusters.

MacQueen (1967) proposed two different methods. The first one, which is the default option in the Quick Cluster procedure of IBM SPSS Statistics (Norušis, 2011), takes the first K points in $\mathcal{X}$ as the centers. An obvious drawback of this method is its sensitivity to data ordering. The second method chooses the centers randomly from the data points. The rationale behind this method is that random selection is likely to pick points from dense regions, i.e. points that are good candidates to be centers. However, there is no mechanism to avoid choosing outliers or points that are too close to each other (Anderberg, 1973). Multiple runs of this method is the standard way of initializing k-means (Bradley & Fayyad, 1998). It should be noted that this second method is often mistakenly attributed to Forgy (1965).

Ball and Hall's method (Ball & Hall, 1967) takes the centroid of $\mathcal{X}$, i.e. $\overline{\mathcal{X}} = 1/N\sum_{j=1}^{N}\mathbf{x}_j$, as the first center. It then traverses the points in arbitrary order and takes a point as a center if it is at least T units apart from the previously selected centers until K centers are obtained. The purpose of the distance threshold T is to ensure that the seed points are well separated. However, it is difficult to decide on an appropriate value for T. In addition, the method is sensitive to data ordering.

The Simple Cluster Seeking method (Tou & Gonzales, 1974) is identical to Ball and Hall's method with the exception that the first point in $\mathcal{X}$ is taken as the first center. This method is used in the FASTCLUS procedure of SAS (SAS Institute Inc., 2009).

Späth's method (Späth, 1977) is similar to Forgy's method with the exception that the points are assigned to the clusters in a cyclical fashion, i.e. the j-th ($j \in \{1,2,\ldots,N\}$) point is assigned to the $(j-1 \bmod K+1)$-th cluster. In contrast to Forgy's method, this method is sensitive to data ordering.

Maximin method (Gonzalez, 1985; Katsavounidis, Kuo, & Zhang, 1994) chooses the first center $\mathbf{c}_1$ arbitrarily and the i-th ($i \in \{2,3,\ldots,K\}$) center $\mathbf{c}_i$ is chosen to be the point that has the greatest minimum-distance to the previously selected centers, i.e. $\mathbf{c}_1,\mathbf{c}_2,\ldots,\mathbf{c}_{i-1}$. This method was originally developed as a 2-approximation to the K-center clustering problem.[1] It should be noted that, motivated by a vector quantization application, Katsavounidis et al.'s variant (Katsavounidis et al., 1994) takes the point with the greatest Euclidean norm as the first center.

Al-Daoud's density-based method (Al-Daoud & Roberts, 1996) first uniformly partitions the data space into M disjoint hypercubes. It then randomly selects $K\,N_m/N$ points from hypercube m ($m \in \{1,2,\ldots,M\}$) to obtain a total of K centers (N_m is the number

[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.

of points in hypercube m). There are two main disadvantages associated with this method. First, it is difficult to decide on an appropriate value for M. Second, the method has a storage complexity of $\mathcal{O}(2^{BD})$, where B is the number of bits allocated to each attribute.

Bradley and Fayyad's method (Bradley & Fayyad, 1998) starts by randomly partitioning the data set into J subsets. These subsets are clustered using k-means initialized by MacQueen's second method producing J sets of intermediate centers each with K points. These center sets are combined into a superset, which is then clustered by k-means J times, each time initialized with a different center set. Members of the center set that give the least SSE are then taken as the final centers.

Pizzuti, Talia, and Vonella (1999) improved upon Al-Daoud's density-based method using a multiresolution grid approach. Their method starts with 2^D hypercubes and iteratively splits these as the number of points they receive increases. Once the splitting phase is completed, the centers are chosen from the densest hypercubes.

The k-means++ method (Arthur & Vassilvitskii, 2007) interpolates between MacQueen's second method and the maximin method. It chooses the first center randomly and the i-th ($i \in \{2,3,\ldots,K\}$) center is chosen to be $\mathbf{x}' \in \mathcal{X}$ with a probability of $\frac{md(\mathbf{x}')^2}{\sum_{j=1}^{N} md(\mathbf{x}_j)^2}$, where $md(\mathbf{x})$ denotes the minimum-distance from a point $\mathbf{x}$ to the previously selected centers. This method yields an $\Theta(\log K)$ approximation to the MSSC problem. The greedy k-means++ method probabilistically selects $\log(K)$ centers in each round and then greedily selects the center that most reduces the SSE. This modification aims to avoid the unlikely event of choosing two centers that are close to each other.

The PCA-Part method (Su & Dy, 2007) uses a divisive hierarchical approach based on PCA (Principal Component Analysis) (Hotelling, 1936). Starting from an initial cluster that contains the entire data set, the method iteratively selects the cluster with the greatest SSE and divides it into two subclusters using a hyperplane that passes through the cluster centroid and is orthogonal to the direction of the principal eigenvector of the covariance matrix. This procedure is repeated until K clusters are obtained. The centers are then given by the centroids of these clusters. The Var-Part method (Su & Dy, 2007) is an approximation to PCA-Part, where the covariance matrix of the cluster to be split is assumed to be diagonal. In this case, the direction of the splitting hyperplane is orthogonal to the coordinate axis with the greatest variance.

Lu et al.'s method (Lu, Tang, Tang, & Yang, 2008) uses a two-phase pyramidal approach. The attributes of each point are first encoded as integers using 2^Q-level quantization, where Q is a resolution parameter. These integer points are considered to be at level *0* of the pyramid. In the bottom-up phase, starting from level *0*, neighboring data points at level k ($k \in \{0,1,\ldots\}$) are averaged to obtain weighted points at level $k+1$ until at least 20 K points are obtained. Data points at the highest level are refined using k-means initialized with the K points with the largest weights. In the top-down phase, starting from the highest level, centers at level $k+1$ are projected onto level k and then used to initialize the k-th level clustering. The top-down phase terminates when level *0* is reached. The centers at this level are then inverse quantized to obtain the final centers. The performance of this method degrades with increasing dimensionality (Lu et al., 2008).

Onoda et al.'s method (Onoda, Sakai, & Yamada, 2012) first calculates K Independent Components (ICs) (Hyvärinen, 1999) of $\mathcal{X}$ and then chooses the i-th ($i \in \{1,2,\ldots,K\}$) center as the point that has the least cosine distance from the i-th IC.

2.2. Loglinear time-complexity initialization methods

Hartigan's method (Hartigan & Wong, 1979) first sorts the points according to their distances to $\overline{\mathcal{X}}$. The i-th ($i \in \{1,2,\ldots,K\}$) center is then chosen to be the $(1 + (i-1)N/K)$-th point. This method is an improvement over MacQueen's first method in that it is invariant to data ordering and is more likely to produce seeds that are well separated. The computational cost of this method is dominated by the complexity of sorting, which is $\mathcal{O}(N \log N)$.

Al-Daoud's variance-based method (Al-Daoud, 2005) first sorts the points on the attribute with the greatest variance and then partitions them into K groups along the same dimension. The centers are then chosen to be the points that correspond to the medians of these groups. Note that this method disregards all attributes but one and therefore is likely to be effective only for data sets in which the variability is mostly on one dimension.

Redmond and Heneghan's method (Redmond & Heneghan, 2007) first constructs a kd-tree of the data points to perform density estimation and then uses a modified maximin method to select K centers from densely populated leaf buckets. The computational cost of this method is dominated by the complexity of kd-tree construction, which is $\mathcal{O}(N \log N)$.

The ROBIN (ROBust INitialization) method (Al Hasan, Chaoji, Salem, & Zaki, 2009) uses a local outlier factor (LOF) (Breunig, Kriegel, Ng, & Sander, 2000) to avoid selecting outlier points as centers. In iteration i ($i \in \{1,2,\ldots,K\}$), the method first sorts the data points in decreasing order of their minimum-distance to the previously selected centers. It then traverses the points in sorted order and selects the first point that has an LOF value close to *1* as the i-th center. The computational cost of this method is dominated by the complexity of sorting, which is $\mathcal{O}(N \log N)$.

2.3. Quadratic-complexity initialization methods

Astrahan's method (Astrahan, 1970) uses two distance thresholds d_1 and d_2. It first calculates the *density* of each point as the number of points within a distance of d_1. The points are sorted in decreasing order by their densities and the highest density point is chosen as the first center. Subsequent centers are chosen in order of decreasing density subject to the condition that each new center be at least at a distance of d_2 from the previously selected centers. This procedure is continued until no more centers can be chosen. Finally, if more than K centers are chosen, hierarchical clustering is used to group the centers until only K of them remain. The main problem with this method is that it is very sensitive to the values of d_1 and d_2. For example, if d_1 is too small there may be many isolated points with zero density whereas if it is too large a few centers will cover the entire data set (Anderberg, 1973).

Lance and Williams (1967) suggested that the output of a hierarchical clustering algorithm can be used to initialize k-means. Despite the fact that such algorithms often have quadratic or higher complexity, this method is highly recommended in the statistics literature (Milligan, 1980) possibly due to the limited size of the data sets in this field.

Kaufman and Rousseeuw's method (Kaufman & Rousseeuw, 1990) takes $\overline{\mathcal{X}}$ as the first center and the i-th ($i \in \{2,3,\ldots,K\}$) center is chosen to be the point that most reduces the SSE. Since pairwise distances between the data points need to be calculated in each iteration, the time complexity of this method is $\mathcal{O}(N^2)$.

Cao, Liang, and Jiang (2009) formalized Astrahan's density-based method within the framework of a neighborhood-based rough set model. In this model, the ε-neighborhood of a point is defined as the set of points within ε distance from it according to a particular distance measure. Based on this neighborhood model,

the concepts of *cohesion* and *coupling* are defined. The former is a measure of the centrality of a point with respect to its neighborhood; whereas the latter is a measure of separation between two neighborhoods. The method first sorts the data points in decreasing order of their *cohesion* and takes the point with the greatest *cohesion* as the first center. It then traverses the points in sorted order and takes the first point that has a *coupling* of less than ε with the previously selected centers as the i-th ($i \in \{2,3,\ldots, K\}$) center. The computational cost of this method is dominated by the complexity of the ε-neighborhood calculations, which is $\mathcal{O}(N^2)$.

2.4. Other initialization methods

The binary-splitting method (Linde, Buzo, & Gray, 1980) takes $\overline{\mathcal{X}}$ as the first center. In iteration t ($t \in \{1,2,\ldots,\log_2 K\}$), each of the existing 2^{t-1} centers is split into two new centers by subtracting and adding a fixed perturbation vector $\boldsymbol{\epsilon}$, i.e. $\mathbf{c}_i - \boldsymbol{\epsilon}$ and $\mathbf{c}_i + \boldsymbol{\epsilon}$ ($i \in \{1,2,\ldots,2^{t-1}\}$). These 2^t new centers are then refined using k-means. There are two main disadvantages associated with this method. First, there is no guidance on the selection of a proper value for $\boldsymbol{\epsilon}$, which determines the direction of the split (Huang & Harris, 1993). Second, the method is computationally demanding since after each iteration k-means has to be run for the entire data set.

The directed-search binary-splitting method (Huang & Harris, 1993) is an improvement over the binary-splitting method in that it determines the value of $\boldsymbol{\epsilon}$ using PCA. However, it has even higher computational requirements due to the calculation of the principal eigenvector in each iteration.

The global k-means method (Likas, Vlassis, & Verbeek, 2003) takes $\overline{\mathcal{X}}$ as the first center. In iteration i ($i \in \{1,2,\ldots,K-1\}$) it considers each of the N points in turn as a candidate for the $(i+1)$-st center and runs k-means with $i+1$ centers on the entire data set. This method is computationally prohibitive for large data sets as it involves $N(K-1)$ runs of k-means on the entire data set.

It should be noted that the two splitting methods and the global k-means method are not initialization methods *per se*. These methods can be considered as complete clustering methods that utilize k-means as a local search procedure. For this reason, to the best of our knowledge, none of the initialization studies to date included these methods in their comparisons.

We should also mention IMs based on metaheuristics such as simulated annealing (Babu & Murty, 1994) and genetic algorithms (Babu & Murty, 1993). These algorithms start from a random initial configuration (population) and use k-means to evaluate their solutions in each iteration (generation). There are two main disadvantages associated with these methods. First, they involve numerous parameters that are difficult to tune (initial temperature, cooling schedule, population size, crossover/mutation probability, etc.) (Jain et al.,, 1999). Second, due to the large search space, they often require a large number of iterations, which renders them computationally prohibitive for all but the smallest data sets. Interestingly, with the recent developments in combinatorial optimization algorithms, it is now feasible to obtain globally minimum SSE clusterings for small data sets without resorting to metaheuristics (Aloise, Hansen, & Liberti, 2010).

2.5. Linear vs. superlinear initialization methods

Based on the descriptions given above, it can be seen that superlinear methods often have more elaborate designs when compared to linear ones. An interesting feature of the superlinear methods is that they are often deterministic, which can be considered as an advantage especially when dealing with large data sets. In contrast, linear methods are often non-deterministic and/or order-sensitive. As a result, it is common practice to perform multiple runs of such methods and take the output of the run that produces the least SSE (Bradley & Fayyad, 1998).

A frequently cited advantage of the more elaborate methods is that they often lead to faster k-means convergence, i.e. require fewer iterations, and as a result the time gained during the clustering phase can offset the time lost during the initialization phase (Su & Dy, 2007; Redmond & Heneghan, 2007; Al Hasan et al., 2009). This may be true when a standard implementation of k-means is used. However, convergence speed may not be as important when a fast k-means variant is used as such methods often require significantly less time compared to a standard k-means implementation. In this study, we utilize a fast k-means variant based on triangle inequality (Huang & Chen, 1990) and partial distance elimination (Bei & Gray, 1985) techniques. As will be seen in Section 4, this fast and exact k-means implementation will diminish the computational efficiency differences among various IMs. In other words, we will demonstrate that elaborate methods that lead to faster k-means convergence are not necessarily more efficient than simple methods with slower convergence.

3. Experimental setup

3.1. Data set descriptions

In order to obtain a comprehensive evaluation of various IMs, we conducted two sets of experiments. The first experiment involved *32* commonly used real data sets with sizes ranging from *214* to *1,904,711* points. Most of these data sets were obtained from the UCI Machine Learning Repository (Frank & Asuncion, 2011) (see Table 1.) The second experiment involved a large number of synthetic data sets with varying clustering complexity. We used a recent algorithm proposed by Maitra and Melnykov (2010) to generate these data sets. This algorithm involves the calculation of the exact overlap (ω) between each cluster pair, measured in terms of their total probability of misclassification, and guided simulation of Gaussian mixture components satisfying prespecified overlap characteristics. The algorithm was used with the following parameters: mean overlap ($\bar{\omega} \in \{0.025, 0.05, 0.1, 0.2\}$), number of points ($N \in \{1024,4096,16384,65536\}$), number of attributes ($D \in \{2,4,8,16,32,64\}$), and number of classes ($K' \in \{2,4,6,8,10,12\}$).

The parameter $\bar{\omega}$ denotes the mean overlap between pairs of clusters. However, we observed that two synthetic data sets with the same $\bar{\omega}$ can have considerably different clustering complexity. Therefore, we quantified clustering complexity using the following indirect approach. For each data set, we executed the k-means algorithm initialized with the "true" centers given by the cluster generation algorithm and calculated the RAND, VD, and VI measures (see Section 3.3) upon convergence. The average of these measures, Ω, was taken as a quantitative indicator of clustering complexity. Note that each of these normalized measures takes values from the [0,1] interval. For RAND larger values are better, whereas for VD and VI smaller values are better. Therefore, we inverted the RAND values by subtracting them from *1* to make this measure compatible with the other two. Finally, using the aforementioned complexity quantification scheme, we generated *4,096* synthetic data sets from each of the following complexity classes: easy ($0 \leqslant \Omega \leqslant 0.25$), moderate ($0.25 < \Omega \leqslant 0.5$), and difficult ($0.5 < \Omega \leqslant 1$). The total number of synthetic data sets was thus $3 \times 4{,}096 = 12{,}288$. Fig. 1 shows sample data sets with $K = 6$ clusters from each complexity class.

3.2. Attribute normalization

In clustering tasks, normalization is a common preprocessing step that is necessary to prevent attributes with large ranges from

Table 1
Descriptions of real data sets.

ID	Data set	# Points (N)	# Attributes (D)	# Classes (K')
1	Breast cancer wisconsin (original)	683	9	2
2	Cloud cover (DB1)	1024	10	8*
3	Concrete compressive strength	1030	9	8*
4	Corel image features	68,040	25	16*
5	Covertype	581,012	10	7
6	Ecoli	336	7	8
7	Steel plates faults	1941	27	7
8	Glass identification	214	9	6
9	Heart disease	297	13	5
10	Ionosphere	351	34	2
11	ISOLET	7797	617	26
12	Landsat satellite (Statlog)	6435	36	6
13	Letter recognition	20,000	16	26
14	MAGIC gamma telescope	19,020	10	2
15	Multiple features (Fourier)	2000	76	10
16	MiniBooNE particle identification	130,064	50	2
17	Musk (Clean2)	6598	166	2
18	Optical digits	5620	64	10
19	Page blocks classification	5473	10	5
20	Parkinsons	5875	18	42*
21	Pen digits	10,992	16	10
22	Person activity	164,860	3	11
23	Pima Indians diabetes	768	8	2
24	Image segmentation	2310	19	7
25	Shuttle (Statlog)	58,000	9	7
26	SPECTF heart	267	44	2
27	Telugu vowels (Pal & Majumder, 1977)	871	3	6
28	Vehicle silhouettes (Statlog)	846	18	4
29	Wall-following robot navigation	5456	24	4
30	Wine quality	6497	11	7
31	World TSP (Cook, 2011)	1,904,711	2	7*
32	Yeast	1484	8	10

* Due to the unavailability of class labels, for data sets #2, #3, and #4, K' was chosen arbitrarily, whereas for #20 and #31, it was determined based on domain knowledge.

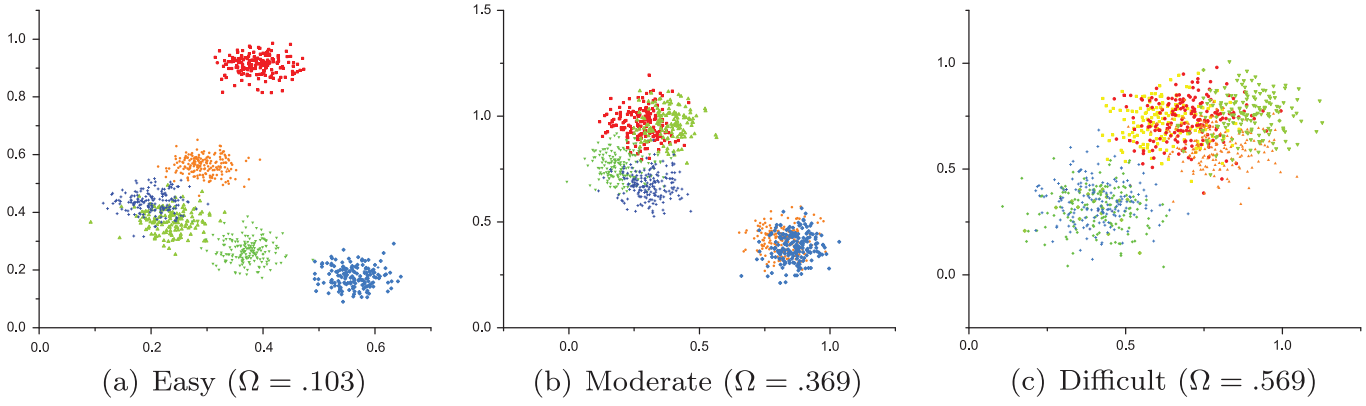

Fig. 1. Synthetic data sets with $K = 6$ clusters.

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 over the latter since the latter is likely to eliminate valuable between-cluster variation (Milligan & Cooper, 1988; Su & Dy, 2007). As a result, we used min–max normalization to map the attributes of each real data set to the [0,1] interval. Note that attributes of the synthetic data sets were already normalized by the cluster generation algorithm.

3.3. Performance criteria

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

- ▷ Initial SSE: 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: 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.

- Normalized Rand (RAND) (Hubert & Arabie, 1985), van Dongen (VD) (van Dongen, 2000), and Variation of Information (Meilă, 2007) criteria (VI): These are external validity measures that quantify the extent to which the clustering structure discovered by a clustering algorithm matches some external structure, e.g. one specified by the given class labels (Wu, Xiong, & Chen, 2009b; Wu, Chen, Xiong, & Xie, 2009a). In a recent comprehensive study, these three measures were found to be the best among 16 external validity measures. Note that each of these normalized measures takes values from the [0,1] interval (Wu et al., 2009b).
- Number of iterations: 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 implementation style, compiler, and CPU architecture.
- CPU time: This is the total CPU time taken by the initialization and clustering phases. This criterion is reported only for the real data sets.

All of the methods were implemented in the C language, compiled with the gcc v4.4.3 compiler, and executed on an Intel Xeon E5520 2.26 GHz machine. Time measurements were performed using the getrusage function, which is capable of measuring CPU time to an accuracy of a microsecond. The MT19937 variant of the Mersenne Twister algorithm was used to generate high-quality pseudorandom numbers (Matsumoto & Nishimura, 1998).

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, i.e. $(\mathrm{SSE}_{i-1} - \mathrm{SSE}_i)/\mathrm{SSE}_i \leqslant \varepsilon$, where SSE_i denotes the SSE value at the end of the i-th ($i \in \{1,2,\ldots,100\}$) iteration. The convergence threshold was set to $\varepsilon = 10^{-6}$.

4. Experimental results and discussion

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. Eight commonly used, order-invariant IMs were included in the experiments: Forgy's method (F), MacQueen's second method (M), maximin (X), Bradley and Fayyad's method (B) with $J = 10$, k-means++ (K), greedy k-means++ (G), Var-Part (V), and PCA-Part (P). It should be noted that among these methods only V and P are deterministic.

In the experiments, each non-deterministic method was executed a 100 times and statistics such as minimum, mean, and standard deviation were collected for the effectiveness criteria. In each run, the number of clusters (K) was set equal to the number of classes (K'), as commonly seen in the related literature (Bradley & Fayyad, 1998; Pizzuti et al., 1999; He et al., 2004; Arthur & Vassilvitskii, 2007; Su & Dy, 2007; Al Hasan et al., 2009; Cao et al., 2009).

Tables 2 and 3 give the Final SSE and CPU time (in milliseconds) results for the real data sets, respectively. Note that, due to space limitations, only mean values are reported for the CPU time criterion. In order to determine if there are any statistically significant differences among the methods, we employed two non-parametric statistical tests (Garcia & Herrera, 2008): the Friedman test (Friedman, 1937) and Iman & Davenport test (Iman & Davenport, 1980). These tests are alternatives to the parametric two-way analysis of variance (ANOVA) test. Their advantage over ANOVA is that they do not require normality or homoscedasticity, assumptions that are often violated in machine learning studies (Luengo, Garcia, & Herrera, 2009; Garcia, Fernandez, Luengo, & Herrera, 2009).

Given B blocks (subjects) and T treatments (measurements), the null hypothesis (H_0) of the Friedman test is that populations within a block are identical. The alternative hypothesis (H_1) is that at least one treatment tends to yield larger (or smaller) values than at least one other treatment. The test statistic is calculated as follows (Daniel, 2000). In the first step, the observations within each block are ranked separately, so each block contains a separate set of T ranks. If ties occur, the tied observations are given the mean of the rank positions for which they are tied. If H_0 is true, the ranks in each block should be randomly distributed over the columns (treatments). Otherwise, we expect a lack of randomness in this distribution. For example, if a particular treatment is better than the others, we expect large (or small) ranks to 'favor' that column. In the second step, the ranks in each column are summed. If H_0 is true, we expect the sums to be fairly close — so close that we can attribute differences to chance. Otherwise, we expect to see at least one difference between pairs of rank sums so large that we cannot reasonably attribute it to sampling variability. The test statistic is given as:

$$\chi_r^2 = \frac{12}{BT(T+1)} \sum_{j=1}^{T} R_j^2 - 3B(T+1) \tag{3}$$

where R_j ($j \in \{1,2,\ldots,T\}$) is the rank sum of the j-th column. χ_r^2 is approximately chi-square with $T-1$ degrees of freedom. H_0 is rejected at the α level of significance if the value of (3) is greater than or equal to the critical chi-square value for $T-1$ degrees of freedom. Iman and Davenport (1980) proposed the following statistic:

$$F_r = \frac{(B-1)\chi_r^2}{B(T-1) - \chi_r^2} \tag{4}$$

which is distributed according to the F-distribution with $T-1$ and $(T-1)(B-1)$ degrees of freedom. When compared to χ_r^2, this statistic is not only less conservative, but also more accurate for small sample sizes (Iman & Davenport, 1980).

In this study, blocks and treatments correspond to data sets and initialization methods, respectively. Our goal is to determine whether or not there is at least one method that is significantly better than at least one other method at the $\alpha = 0.05$ level. If this is the case, we will conduct a post hoc (multiple comparison) test to determine which pairs of methods differ significantly. For this purpose, we will use the Bergmann–Hommel test (Bergmann & Hommel, 1988), a powerful post hoc procedure that has been used successfully in various machine learning studies (Garcia & Herrera, 2008; Voss, Hansen, & Igel, 2010).

4.1. Real data sets

Table 4 gives the Final SSE rankings of the IMs for the real data sets as determined by the Bergmann–Hommel procedure using data given in Table 2. Here, a notation such as C < {D,E} indicates that there is no statistically significant difference between methods D and E and these two methods are significantly better than method C. From Table 4 it can be seen that the methods cannot be distinguished from each other reliably. This was expected since even nonparametric post hoc tests lack discrimination power in small sample cases (recall that only 32 data sets were used) with a large number of ties (see Table 2). For example, with respect to the *minimum* statistic, the performances of F, M, B, K, and G are statistically indistinguishable. In other words, if we initialize k-means with each of these non-deterministic methods and execute it until convergence, the resulting clusterings over $R = 100$ runs will have very similar *minimum* Final SSE values. Similar trends were observed for the RAND, VD, and VI criteria (data not shown). Given

Table 2
Final SSE (real data sets).

		F	M	X	B	K	G	V	P
1	min	239	239	239	239	239	239	239	239
	mean	239 ± 0	239 ± 0	239 ± 0	239 ± 0	239 ± 0	239 ± 0	239 ± 0	239 ± 0
2	min	38	38	41	38	38	38	39	38
	mean	44 ± 4	40 ± 1	45 ± 3	39 ± 1	39 ± 1	39 ± 1	39 ± 0	38 ± 0
3	min	167	167	167	167	167	167	173	167
	mean	176 ± 8	176 ± 7	172 ± 6	171 ± 4	174 ± 5	173 ± 4	173 ± 0	167 ± 0
4	min	10057	10057	10057	10057	10058	10058	10101	10100
	mean	10115 ± 79	10080 ± 20	10077 ± 15	10076 ± 18	10083 ± 23	10080 ± 18	10101 ± 0	10100 ± 0
5	min	66224	66224	66224	66224	66224	66224	66238	66238
	mean	66990 ± 890	67196 ± 1048	67350 ± 834	66431 ± 360	66948 ± 876	66930 ± 773	66238 ± 0	66238 ± 0
6	min	17	17	19	17	17	17	17	18
	mean	19 ± 2	19 ± 3	20 ± 1	18 ± 1	18 ± 1	18 ± 1	17 ± 0	18 ± 0
7	min	1167	1167	1267	1167	1167	1167	1167	1168
	mean	1231 ± 83	1250 ± 103	1303 ± 53	1184 ± 25	1230 ± 61	1198 ± 35	1167 ± 0	1168 ± 0
8	min	18	18	19	18	18	18	19	19
	mean	20 ± 1	20 ± 2	22 ± 2	20 ± 1	20 ± 2	20 ± 1	19 ± 0	19 ± 0
9	min	243	243	243	243	243	243	248	243
	mean	251 ± 8	252 ± 8	253 ± 8	251 ± 7	252 ± 8	249 ± 7	248 ± 0	243 ± 0
10	min	629	629	629	629	629	629	629	629
	mean	629 ± 0	633 ± 28	671 ± 81	637 ± 39	635 ± 34	635 ± 35	629 ± 0	629 ± 0
11	min	117891	117924	120898	117863	117719	117995	118495	118386
	mean	119931 ± 1060	119655 ± 1061	123388 ± 1264	119050 ± 699	119538 ± 894	119176 ± 710	118495 ± 0	118386 ± 0
12	min	1742	1742	1742	1742	1742	1742	1742	1742
	mean	1742 ± 0	1742 ± 0	1742 ± 0	1742 ± 0	1744 ± 28	1747 ± 41	1742 ± 0	1742 ± 0
13	min	2723	2718	2721	2719	2718	2715	2735	2745
	mean	2775 ± 28	2756 ± 22	2765 ± 17	2742 ± 15	2754 ± 18	2752 ± 17	2735 ± 0	2745 ± 0
14	min	2923	2923	2923	2923	2923	2923	2923	2923
	mean	2923 ± 0	2923 ± 0	2923 ± 0	2923 ± 0	2923 ± 0	2923 ± 0	2923 ± 0	2923 ± 0
15	min	3127	3128	3180	3127	3128	3127	3137	3214
	mean	3164 ± 30	3168 ± 28	3247 ± 22	3157 ± 29	3173 ± 33	3149 ± 20	3137 ± 0	3214 ± 0
16	min	2802	2802	2802	2802	2802	2802	21983	2802
	mean	8229 ± 8685	12518 ± 9667	2802 ± 0	11935 ± 9656	5722 ± 6944	3774 ± 4236	21983 ± 0	2802 ± 0
17	min	36373	36373	36373	36373	36373	36373	36373	36373
	mean	37755 ± 2829	37046 ± 916	36738 ± 754	37152 ± 1340	37440 ± 1906	37103 ± 1639	36373 ± 0	36373 ± 0
18	min	14559	14559	14559	14559	14559	14559	14581	14807
	mean	14653 ± 140	14763 ± 273	14774 ± 293	14627 ± 66	14735 ± 234	14719 ± 214	14581 ± 0	14807 ± 0
19	min	215	215	230	215	215	215	227	215
	mean	217 ± 4	217 ± 4	254 ± 32	219 ± 6	219 ± 10	217 ± 4	227 ± 0	215 ± 0
20	min	235	219	233	218	217	217	220	219
	mean	251 ± 7	224 ± 2	241 ± 3	222 ± 2	220 ± 2	219 ± 1	220 ± 0	219 ± 0
21	min	4930	4930	4930	4930	4930	4930	4930	5004
	mean	5130 ± 131	5091 ± 110	5036 ± 106	5012 ± 70	5111 ± 116	5046 ± 75	4930 ± 0	5004 ± 0
22	min	1177	1177	1195	1177	1177	1177	1182	1177
	mean	1179 ± 10	1187 ± 18	1204 ± 25	1182 ± 12	1193 ± 27	1183 ± 14	1182 ± 0	1177 ± 0
23	min	121	121	121	121	121	121	121	121
	mean	121 ± 2	122 ± 5	122 ± 3	122 ± 3	122 ± 5	122 ± 5	121 ± 0	121 ± 0
24	min	387	387	411	387	387	387	410	405
	mean	407 ± 23	414 ± 20	430 ± 21	402 ± 16	410 ± 19	402 ± 13	410 ± 0	405 ± 0
25	min	235	235	411	235	235	235	235	274
	mean	307 ± 39	275 ± 23	930 ± 105	244 ± 18	271 ± 39	246 ± 21	235 ± 0	274 ± 0
26	min	214	214	214	214	214	214	214	214
	mean	214 ± 0	214 ± 0	214 ± 0	214 ± 0	214 ± 0	214 ± 0	214 ± 0	214 ± 0
27	min	22	22	22	22	22	22	23	23
	mean	23 ± 2	23 ± 1	23 ± 0	23 ± 1	23 ± 1	23 ± 0	23 ± 0	23 ± 0
28	min	223	223	224	223	223	223	224	224
	mean	224 ± 2	226 ± 4	237 ± 1	228 ± 6	226 ± 5	225 ± 3	224 ± 0	224 ± 0
29	min	7772	7772	7772	7772	7772	7772	7774	7774
	mean	7798 ± 91	7808 ± 102	7854 ± 160	7773 ± 1	7831 ± 140	7811 ± 106	7774 ± 0	7774 ± 0
30	min	334	334	348	334	334	334	335	334
	mean	335 ± 2	336 ± 2	374 ± 17	337 ± 5	336 ± 3	336 ± 3	335 ± 0	334 ± 0

Table 2 (*continued*)

		F	M	X	B	K	G	V	P
31	min	11039	11039	11039	11039	11039	11039	11483	12422
	mean	14041 ± 1686	12367 ± 1057	11714 ± 627	11128 ± 231	11773 ± 872	11493 ± 626	11483 ± 0	12422 ± 0
32	min	58	58	61	58	58	58	69	59
	mean	64 ± 5	70 ± 6	61 ± 1	66 ± 6	63 ± 5	59 ± 1	69 ± 0	59 ± 0

Table 3
CPU time (real data sets).

	F	M	X	B	K	G	V	P
1	0	0	0	0	0	0	0	0
2	3	3	2	4	3	2	10	0
3	2	2	2	4	2	2	0	10
4	2295	2248	2173	3624	2332	2459	1900	2540
5	2183	2229	2714	3604	2273	2274	1730	2120
6	0	0	0	1	0	0	0	0
7	8	8	7	12	9	9	0	20
8	0	0	0	1	0	0	0	0
9	0	1	0	1	0	1	0	0
10	0	0	0	1	0	0	0	0
11	3730	3469	2469	4063	3537	3915	6940	12200
12	28	32	40	40	34	32	40	50
13	700	693	729	852	698	693	950	800
14	20	19	28	30	21	19	30	20
15	54	56	62	70	57	58	30	70
16	252	283	58	417	112	96	230	570
17	22	20	24	34	25	26	20	220
18	112	116	131	137	121	125	60	140
19	9	10	7	12	8	10	10	10
20	109	100	110	150	97	118	60	70
21	59	53	52	67	56	59	30	50
22	430	524	314	718	469	513	730	480
23	1	1	1	1	1	0	0	0
24	6	5	7	8	6	7	10	0
25	84	82	19	122	79	81	100	80
26	0	0	0	1	0	0	10	0
27	1	1	0	1	1	1	0	0
28	3	2	1	3	2	3	10	10
29	20	20	20	28	19	21	10	20
30	38	38	24	51	36	38	60	40
31	748	949	1377	2439	918	1044	580	840
32	5	6	5	9	6	6	0	0

Table 4
Final SSE rankings (real data sets).

Statistic	IM ranking
Minimum	{X,V,P} < {F,M,B,K,G}
Mean	{F,M,X,K} < {F,B,G} < {G,V} < {V,P}
Standard deviation	{F,M,X,B,K} < {X,B,G}

the abundance of local minima even in data sets of moderate size and/or dimensionality and the gradient descent nature of k-means, it is not surprising that the deterministic methods V and P are outperformed by most of the non-deterministic methods as the former methods were executed only once, whereas the latter ones were executed $R = 100$ times.

As mentioned earlier, the *minimum* statistic is meaningful only when it is practical to execute k-means multiple times. Otherwise, the *mean* statistic is more meaningful. The analysis of *mean* Final SSE results using the Bergmann–Hommel procedure reveals that deterministic methods V and P are the preferred choices in this case. This is not surprising since non-deterministic methods, in particular those that are *ad hoc* in nature, often produce highly variable results across multiple runs.

The *standard deviation* statistic characterizes the reliability of a non-deterministic IM with respect to a particular performance criterion. In other words, if a non-deterministic IM obtains low *mean* and *standard deviation* with respect to an effectiveness criterion, we do not have to execute this method a large number of times to obtain good results. The analysis of Final SSE *standard deviations* reveals two overlapping groups of methods. Once again this is not necessarily because the members of each group are in fact indistinguishable with respect to their reliability, but due to the relatively small sample size used. In summary, due to the necessarily small number of real-world data sets available for clustering studies, it may not be possible to distinguish among various IMs. Therefore, it is crucial that these IMs be tested on a large number of synthetic data sets (see Section 4.2).

As for computational efficiency, it can be seen from Table 3 that, in general, the IMs have similar computational requirements per run. However, in practice, a non-deterministic method is typically executed R times and the output of the run that gives the least SSE is taken as the result. Therefore, the total computational cost of a non-deterministic method is often significantly higher than that of a deterministic method. As predicted in Section 2.5, simple methods such as M require about the same CPU time as elaborate methods such as G. This is because simple methods often lead to more k-means iterations, whereas elaborate ones compensate for their computational overhead by requiring fewer k-means iterations. It should be noted that efficiency differences among the methods can be further reduced by using faster k-means variants such as those described in Kanungo et al. (2002) and Hamerly (2010).

Table 5
Minimum (synthetic data sets).

Data set complexity	IM ranking
Initial SSE	
Easy	X < {F,M} < K < G < V < P < B
Moderate	X < {F,M,K} < G < V < P < B
Difficult	X < {M,K} < F < G < V < P < B
Final SSE	
Easy	{V,P} < {F,X} < {M,B,K,G}
Moderate	{V,P} < {F,X} < {F,M,B,K} < {M,B,K,G}
Difficult	V < P < {F,X} < {M,X,B,K,G}
Final RAND	
Easy	{V,P} < X < F < {M,K} < G < B
Moderate	{V,P} < X < {F,M,K} < {M,K,G} < {B,G}
Difficult	V < P < {F,X} < {F,K,G} < {M,K,G} < {M,B,K}
Final VD	
Easy	{V,P} < X < {F,M} < {K,G} < B
Moderate	{V,P} < X < {F,M} < {M,K} < {B,K,G}
Difficult	V < P < {F,X,G} < {M,K,G} < {M,B,K}
Final VI	
Easy	{V,P} < X < {F,M,K,G} < B
Moderate	{V,P} < X < F < {M,B,K,G}
Difficult	V < P < {F,X} < {X,G} < {M,B,K,G}
Number of iterations	
Easy	V < F < M < {X,K} < P < G < B
Moderate	V < P < F < M < K < {X,G} < B
Difficult	V < P < F < M < K < G < X < B

4.2. Synthetic data sets

Table 5 gives the ranking of the IMs with respect to the *minimum* statistic. It can be seen that despite variations in rankings across the performance criteria, some general trends emerge:

- Non-deterministic methods outperform the deterministic ones, i.e. V and P, except in the case of Initial SSE. As explained in Section 4.1, this is due to the fact that the non-deterministic methods were executed $R = 100$ times, whereas the deterministic ones were executed only once. The reason why deterministic methods have good Initial SSE performance is because these methods are approximate (divisive hierarchical) clustering methods by themselves and thus they give reasonable results even without k-means refinement.
- Method B consistently appears in the best performing group, whereas methods F and X are often among the worst non-deterministic methods.
- Method M exhibits moderate-to-good performance except in the case of Initial SSE. Recall that this method randomly selects the K initial centers from among the data points and therefore it cannot be expected to perform well without k-means refinement.
- Methods K and G generally perform well. In some cases the latter outperforms the former, whereas in others they have comparable performance.

Table 6 gives the ranking of the IMs with respect to the *mean* statistic. It can be seen that despite variations in rankings across the performance criteria, some general trends emerge:

- Deterministic methods, i.e. V and P, generally outperform the non-deterministic ones. As explained in Section 4.1, this is due to the fact that the non-deterministic methods can produce highly variable results across multiple runs. Method B is highly competitive with the deterministic methods.
- Methods M and X are often among the worst performers, whereas methods F and K exhibit moderate-to-bad performance.
- Method G is often significantly better than all non-deterministic methods but B.

Table 6
Mean (synthetic data sets).

Data set complexity	IM ranking
Initial SSE	
Easy	X < M < K < F < G < V < {B,P}
Moderate	X < {M,K} < F < G < V < {B,P}
Difficult	X < K < M < F < G < V < B < P
Final SSE	
Easy	{M,X} < K < F < G < B < {V,P}
Moderate	X < {M,K} < F < G < B < V < P
Difficult	F < {M,K} < X < G < B < V < P
Final RAND	
Easy	{M,X} < K < F < G < B < {V,P}
Moderate	X < {M,K} < {F,G} < B < {V,P}
Difficult	{F,K} < {X,K,G} < M < V < B < P
Final VD	
Easy	{M,X} < K < F < G < B < {V,P}
Moderate	X < {M,K} < F < G < B < V < P
Difficult	F < {M,X,K,G} < V < B < P
Final VI	
Easy	{M,X} < K < F < G < B < {V,P}
Moderate	{M,X,K} < F < G < B < V < P
Difficult	F < {M,K,G} < {X,G} < V < B < P
Number of iterations	
Easy	M < X < K < F < G < V < P < B
Moderate	{F,M} < {X,K} < G < V < P < B
Difficult	F < M < K < G < X < V < P < B

Table 7
Standard deviation (synthetic data sets).

Data set complexity	IM ranking
Initial SSE	
Easy	X < M < K < G < F < B
Moderate	Same as easy
Difficult	X < M < K < G < {F,B}
Final SSE	
Easy	M < {X,K} < G < F < B
Moderate	{M,K} < X < {F,G} < B
Difficult	{M,K} < {F,X,G} < B
Final RAND	
Easy	M < {X,K} < G < F < B
Moderate	{M,X,K} < {F,G} < B
Difficult	{M,K} < {F,X,G} < B
Final VD	
Easy	M < {X,K} < G < F < B
Moderate	{M,K} < X < {F,G} < B
Difficult	{M,K} < {K,G} < {F,G} < X < B
Final VI	
Easy	M < {X,K} < {F,G} < B
Moderate	{M,K} < X < {F,G} < B
Difficult	{F,M,K,G} < X < B
Number of iterations	
Easy	M < K < {X,G} < F < B
Moderate	{M,K} < X < {F,G} < B
Difficult	{F,M,X,K,G} < B

Table 7 gives the ranking of the non-deterministic IMs with respect to the *standard deviation* statistic. It can be seen that despite variations in rankings across the performance criteria, some general trends emerge:

- Method B consistently appears in the best performing group, whereas method M is often among the worst performers.
- Methods X and K exhibit moderate-to-bad performance.
- Method F and G are significantly better than all methods but B.

4.3. Recommendations for practitioners

Based on the statistical analyses presented in the previous section, the following recommendations can be made:

- In general, methods F, M, and X should not be used. These methods are easy to understand and implement, but they are often ineffective and unreliable. Furthermore, despite their low overhead, these methods do not offer significant time savings since they often result in slower k-means convergence.
- In time-critical applications that involve large data sets or applications that demand determinism, methods V or P should be used. These methods need to be executed only once and they lead to very fast k-means converge. The efficiency difference between the two is noticeable only on high dimensional data sets. This is because method V calculates the direction of split by determining the coordinate axis with the greatest variance (in $\mathcal{O}(D)$ time), whereas method P achieves this by calculating the principal eigenvector of the covariance matrix (in $\mathcal{O}(D^2)$ time using the power method (Hotelling, 1936)). Note that despite its higher computational complexity, method P can, in some cases, be more efficient than method V (see Table 3). This

is because the former converges significantly faster than the latter (see Table 6). The main disadvantage of these methods is that they are more complicated to implement due to their hierarchical formulation.

- In applications that involve small data sets, e.g. $N < 10{,}000$, methods B or G should be used. It is computationally feasible to run these methods hundreds of times on such data sets given that one such run takes only a few milliseconds.
- In applications where an approximate clustering of the data set is desired, methods B, G, V, or P should be used. These methods produce very good initial clusterings (see Tables 5 and 6), which makes it possible to use them as standalone clustering algorithms.

5. Conclusions

In this paper we presented an overview of k-means initialization methods with an emphasis on their computational efficiency. We then compared eight commonly used linear time initialization methods on a large and diverse collection of real and synthetic data sets using various performance criteria. Finally, we analyzed the experimental results using non-parametric statistical tests and provided recommendations for practitioners. Our statistical analyses revealed that popular initialization methods such as *forgy*, *macqueen*, and *maximin* often perform poorly and that there are significantly better alternatives to these methods that have comparable computational requirements.

Acknowledgments

This publication was made possible by grants from the Louisiana Board of Regents (LEQSF2008-11-RD-A-12) and National Science Foundation (0959583, 1117457). The authors are grateful to D. Arthur, J.G. Dy, S.J. Redmond, J.F. Lu, and M. Al Hasan for clarifying various points about their papers.

References

Al-Daoud, M. (2005). A new algorithm for cluster initialization, In: *Proc. of the 2nd World Enformatika conf.* (pp. 74–76).

Al-Daoud, M., & Roberts, S. (1996). New methods for the initialisation of clusters. *Pattern Recognition Letters, 17*(5), 451–455.

Al Hasan, M., Chaoji, V., Salem, S., & Zaki, M. (2009). Robust partitional clustering by outlier and density insensitive seeding. *Pattern Recognition Letters, 30*(11), 994–1002.

Aloise, D., Deshpande, A., Hansen, P., & Popat, P. (2009). NP-hardness of euclidean sum-of-squares clustering. *Machine Learning, 75*(2), 245–248.

Aloise, D., Hansen, P., & Liberti, L. (2010). An improved column generation algorithm for minimum sum-of-squares clustering. *Mathematical Programming*, 1–26.

Anderberg, M. R. (1973). *Cluster analysis for applications*. Academic Press.

Arthur, D., & Vassilvitskii, S. (2007). k-means++: The advantages of careful seeding, In: *Proc. of the 18th annual ACM-SIAM symposium on discrete algorithms* (pp. 1027–1035).

Astrahan, M. M. (1970). Speech analysis by clustering, or the hyperphoneme method, Tech. Rep. AIM-124, Stanford University.

Babu, G. P., & Murty, M. N. (1993). A near-optimal initial seed value selection in k-means algorithm using a genetic algorithm. *Pattern Recognition Letters, 14*(10), 763–769.

Babu, G., & Murty, M. (1994). Simulated annealing for selecting optimal initial seeds in the k-means algorithm. *Indian Journal of Pure and Applied Mathematics, 25*(1–2), 85–94.

Ball, G. H., & Hall, D. J. (1967). A clustering technique for summarizing multivariate data. *Behavioral Science, 12*(2), 153–155.

Bei, C. D., & Gray, R. M. (1985). An improvement of the minimum distortion encoding algorithm for vector quantization. *IEEE Transactions on Communications, 33*(10), 1132–1133.

Bergmann, B., & Hommel, G. (1988). *Multiple hypotheses testing. Ch. improvements of general multiple test procedures for redundant systems of hypotheses*. Springer, pp. 100–115.

Bottou, L., & Bengio, Y. (1995). *Advances in neural information processing systems. Ch. convergence properties of the k-means algorithms*. MIT Press, pp. 585–592.

Bradley, P. S., & Fayyad, U. (1998). Refining initial points for k-means clustering, In: *Proc. of the 15th int. conf. on machine learning* (pp. 91–99).

Breunig, M. M., Kriegel, H. P., Ng, R. T., & Sander, J. (2000). LOF: Identifying density-based local outliers. *ACM SIGMOD Record, 29*(2), 93–104.

Cao, F., Liang, J., & Jiang, G. (2009). An initialization method for the k-means algorithm using neighborhood model. *Computers and Mathematics with Applications, 58*(3), 474–483.

Celebi, M. E. (2011). Improving the performance of k-means for color quantization. *Image and Vision Computing, 29*(4), 260–271.

Chen, T. W., & Chien, S. Y. (2010). Bandwidth adaptive hardware architecture of k-means clustering for video analysis. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 18*(6), 957–966.

Chen, W. Y., Song, Y., Bai, H., Lin, C. J., & Chang, E. Y. (2011). Parallel spectral clustering in distributed systems. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 33*(3), 568–586.

Cook, W. (2011). World TSP, Georgia Institute of Technology, <http://www.tsp.gatech.edu/world>.

Daniel, W. W. (2000). *Applied nonparametric statistics*. Duxbury Press.

Dash, M., Liu, H., & Xu, X. (2001). '1 + 1 > 2': Merging distance and density based clustering, In: *Proc. of the 7th int. conf. on database systems for advanced applications* (pp. 32–39).

Forgy, E. (1965). Cluster analysis of multivariate data: Efficiency vs. interpretability of classification. *Biometrics, 21*, 768.

Frank, A., & Asuncion, A. (2011). UCI machine learning repository, University of California, Irvine, School of Information and Computer Sciences, <http://archive.ics.uci.edu/ml>.

Friedman, M. (1937). The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *Journal of the American Statistical Association, 32*(200), 675–701.

Garcia, S., Fernandez, A., Luengo, J., & Herrera, F. (2009). A study of statistical techniques and performance measures for genetics-based machine learning: Accuracy and interpretability. *Soft Computing, 13*(10), 959–977.

Garcia, S., & Herrera, F. (2008). An extension on "statistical comparisons of classifiers over multiple data sets" for all pairwise comparisons. *Journal of Machine Learning Research, 9*, 2677–2694.

Gonzalez, T. (1985). Clustering to minimize the maximum intercluster distance. *Theoretical Computer Science, 38*(2–3), 293–306.

Hamerly, G. (2010). Making k-means even faster, in: *Proc. of the 2010 SIAM int. conf. on data mining* (pp. 130–140).

Hartigan, J. A., & Wong, M. A. (1979). Algorithm AS 136: A k-means clustering algorithm. *Journal of the Royal Statistical Society C, 28*(1), 100–108.

He, J., Lan, M., Tan, C. L., Sung, S. Y., & Low, H. B. (2004). Initialization of cluster refinement algorithms: A review and comparative study, In: *Proc. of the 2004 IEEE int. joint conf. on neural networks* (pp. 297–302).

Hotelling, H. (1936). Simplified calculation of principal components. *Psychometrika, 1*(1), 27–35.

Huang, S. H., & Chen, S. H. (1990). Fast encoding algorithm for VQ-based image coding. *Electronics Letters, 26*(19), 1618–1619.

Huang, C. M., & Harris, R. W. (1993). A comparison of several vector quantization codebook generation approaches. *IEEE Transactions on Image Processing, 2*(1), 108–112.

Hubert, L., & Arabie, P. (1985). Comparing partitions. *Journal of Classification, 2*(1), 193–218.

Hyvärinen, A. (1999). Fast and robust fixed-point algorithms for independent component analysis. *IEEE Transactions on Neural Networks, 10*(3), 626–634.

Iman, R. L., & Davenport, J. M. (1980). Approximations of the critical region of the friedman statistic. *Communications in Statistics – Theory and Methods, 9*(6), 571–595.

Jain, A. K. (2010). Data clustering: 50 years beyond k-means. *Pattern Recognition Letters, 31*(8), 651–666.

Jain, A. K., Murty, M. N., & Flynn, P. J. (1999). Data clustering: A review. *ACM Computing Surveys, 31*(3), 264–323.

Jancey, R. C. (1966). Multidimensional group analysis. *Australian Journal of Botany, 14*(1), 127–130.

Kanungo, T., Mount, D., Netanyahu, N., Piatko, C., Silverman, R., & Wu, A. (2002). An efficient k-means clustering algorithm: analysis and implementation. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 24*(7), 881–892.

Katsavounidis, I., Kuo, C.-C. J., & Zhang, Z. (1994). A new initialization technique for generalized Lloyd iteration. *IEEE Signal Processing Letters, 1*(10), 144–146.

Kaufman, L., & Rousseeuw, P. (1990). *Finding groups in data: An introduction to cluster analysis*. Wiley Interscience.

Lance, G. N., & Williams, W. T. (1967). A general theory of classificatory sorting strategies - II. Clustering systems. *The Computer Journal, 10*(3), 271–277.

Likas, A., Vlassis, N., & Verbeek, J. (2003). The global k-means clustering algorithm. *Pattern Recognition, 36*(2), 451–461.

Linde, Y., Buzo, A., & Gray, R. (1980). An algorithm for vector quantizer design. *IEEE Transactions on Communications, 28*(1), 84–95.

Lloyd, S. (1982). Least squares quantization in PCM. *IEEE Transacations on Information Theory, 28*(2), 129–136.

Luengo, J., Garcia, S., & Herrera, F. (2009). A study on the use of statistical tests for experimentation with neural networks: Analysis of parametric test conditions and non-parametric tests. *Expert Systems with Applications, 36*(4), 7798–7808.

Lu, J. F., Tang, J. B., Tang, Z. M., & Yang, J. Y. (2008). Hierarchical initialization approach for k-means clustering. *Pattern Recognition Letters, 29*(6), 787–795.

MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations, In: *Proc. of the 5th Berkeley symposium on mathematical statistics and probability* (pp. 281–297).

Mahajan, M., Nimbhorkar, P., & Varadarajan, K. (2012). The planar k-means problem is NP-hard. *Theoretical Computer Science*, *442*, 13-21.

Maitra, R., & Melnykov, V. (2010). Simulating data to study performance of finite mixture modeling and clustering algorithms. *Journal of Computational and Graphical Statistics, 19*(2), 354–376.

Mao, J., & Jain, A. K. (1996). A self-organizing network for hyperellipsoidal clustering (HEC). *IEEE Transacations on Neural Networks, 7*(1), 16–29.

Matsumoto, M., & Nishimura, T. (1998). Mersenne twister: A 623-dimensionally equidistributed uniform pseudo-random number generator. *ACM Transactions on Modeling and Computer Simulation, 8*(1), 3–30.

Meilă, M. (2007). Comparing clusterings — An information based distance. *Journal of Multivariate Analysis, 98*(5), 873–895.

Milligan, G. W. (1980). An examination of the effect of six types of error perturbation on fifteen clustering algorithms. *Psychometrika, 45*(3), 325–342.

Milligan, G., & Cooper, M. C. (1988). A study of standardization of variables in cluster analysis. *Journal of Classification, 5*(2), 181–204.

Norušis, M. J. (2011). *IBM SPSS statistics 19 statistical procedures companion*. Addison Wesley.

Onoda, T., Sakai, M., & Yamada, S. (2012). Careful seeding method based on independent components analysis for k-means clustering. *Journal of Emerging Technologies in Web Intelligence, 4*(1), 51–59.

Ordonez, C., & Omiecinski, E. (2004). Efficient disk-based k-means clustering for relational databases. *IEEE Transactions on Knowledge and Data Engineering, 16*(8), 909–921.

Pal, S. K., & Majumder, D. D. (1977). Fuzzy sets and decision making approaches in vowel and speaker recognition. *IEEE Transacations on Systems, Man, and Cybernetics, 7*(8), 625–629 <http://www.isical.ac.in/~sushmita/patterns/vowel.dat>.

Pena, J. M., Lozano, J. A., & Larranaga, P. (1999). An empirical comparison of four initialization methods for the k-means algorithm. *Pattern Recognition Letters, 20*(10), 1027–1040.

Pizzuti, C., Talia, D., & Vonella, G. (1999). A divisive initialisation method for clustering algorithms, In: *Proc. of the 3rd European conf. on principles and practice of knowledge discovery in databases* (pp. 484–491).

Redmond, S. J., & Heneghan, C. (2007). A method for initialising the k-means clustering algorithm using kd-trees. *Pattern Recognition Letters, 28*(8), 965–973.

SAS Institute Inc., SAS/STAT 9.2 User's Guide, SAS Publishing, 2009.

Selim, S. Z., & Ismail, M. A. (1984). K-means-type algorithms: A generalized convergence theorem and characterization of local optimality. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 6*(1), 81–87.

Späth, H. (1977). Computational experiences with the exchange method: Applied to four commonly used partitioning cluster analysis criteria. *European Journal of Operational Research, 1*(1), 23–31.

Su, T., & Dy, J. G. (2007). In search of deterministic methods for initializing k-means and Gaussian mixture clustering. *Intelligent Data Analysis, 11*(4), 319–338.

Tarsitano, A. (2003). A computational study of several relocation methods for k-means algorithms. *Pattern Recognition, 36*(12), 2955–2966.

Tou, J. T., & Gonzales, R. C. (1974). *Pattern recognition principles*. Addison-Wesley.

van Dongen, S. (2000). Performance criteria for graph clustering and Markov cluster experiments, Tech. Rep. INS-R0012, Centrum voor Wiskunde en Informatica.

Voss, T., Hansen, N., & Igel, C. (2010). Improved step size adaptation for the MO-CMA-ES, In: *Proc. of the 12th annual conf. on genetic and evolutionary computation* (pp. 487–494).

Wu, J., Chen, J., Xiong, H., & Xie, M. (2009a). External validation measures for k-means clustering: A data distribution perspective. *Expert Systems with Applications, 36*(3), 6050–6061.

Wu, J., Xiong, H., & Chen, J. (2009b). Adapting the right measures for k-means clustering, In: *Proc. of the 15th ACM SIGKDD int. conf. on knowledge discovery and data mining* (pp. 877–885).

Zhang, J. S., & Leung, Y.-W. (2003). Robust clustering by pruning outliers. *IEEE Transactions on Systems, Man, and Cybernetics – Part B, 33*(6), 983–999.