# A Short Survey on Data Clustering Algorithms

**Article** · November 2015

DOI: 10.1109/ISCMI.2015.10



**1 author:**

Ka-Chun Wong
City University of Hong Kong
**285** PUBLICATIONS **4,998** CITATIONS

# A Short Survey on Data Clustering Algorithms

Ka-Chun Wong
Department of Computer Science
City University of Hong Kong
Kowloon Tong, Hong Kong
Email: kc.w@cityu.edu.hk



***Abstract*—With rapidly increasing data, clustering algorithms are important tools for data analytics in modern research. They have been successfully applied to a wide range of domains; for instance, bioinformatics, speech recognition, and financial analysis. Formally speaking, given a set of data instances, a clustering algorithm is expected to divide the set of data instances into the subsets which maximize the intra-subset similarity and inter-subset dissimilarity, where a similarity measure is defined beforehand. In this work, the state-of-the-arts clustering algorithms are reviewed from design concept to methodology; Different clustering paradigms are discussed. Advanced clustering algorithms are also discussed. After that, the existing clustering evaluation metrics are reviewed. A summary with future insights is provided at the end.**

## I. Introduction

Nowadays, with the support of science and technology, large amounts of data has been, and will continue to be, accumulated. For example, a single human genome accounts for about four gigabytes data space [1], [2], [3] and the transaction logs in financial markets are measured in billions each day [4]. Such a large amount of data is overwhelming and prevents us from applying traditional analysis techniques. Scalable methods need to be devised to handle it. As one of the main analysis tools, cluster analysis methods have been proposed to separate the large amount of data into clusters. The data clustering methods are unsupervised which means there is not any label for model training; we do not even know the exact number of clusters beforehand. Given a set of data, a clustering method is expected to divide the data into several clusters by itself. Formally speaking, given a set of data instances, a data clustering method is expected to divide the set of data instances into the subsets which maximize the intra-subset similarity and inter-subset dissimilarity, where a similarity measure is defined beforehand.

## II. Clustering Paradigms

Since most data clustering problems have been shown to be NP-hard [5], different methods have been proposed in the past. In general, those methods can be categorized into different paradigms: Partitional Clustering, Hierarchical Clustering, Density-based Clustering, Grid-based Clustering, Correlation Clustering, Spectral Clustering, Gravitational Clustering, Herd Clustering, and Others.

### A. Partitional Clustering

Data is divided into non-overlapping subsets such that each data instance is assigned to exactly one subset. For example, k-means [6] is a classical partitioning method that applies an iterative refinement approach with two main steps. The first step is to choose the means of clusters as the centroids, whereas the second step is to assign data points to their nearest centroids. In practice, its computational speed and simplicity appeal to people [7], [8]. Its main drawback is the vulnerability to its random seeding technique. In other words, if the initial seeding positions are not chosen correctly, the clustering result quality will be affected adversely.

In light of that, David Arthur and Sergei Vassilvitskii proposed a method called k-means++ [9] to improve k-means in 2007. From section 2.1.and 2.2 in [9], we can observe that the steps 2-4 of k-means++ are exactly the same as those of k-means. The main difference lies in the step 1 which is the seeding technique. A new seeding technique is proposed to replace the arbitrary seeding technique of k-mean. Given a set of seeds chosen, the seeding technique favors the data points which are far from the seeds already chosen. Thus the seeds are chosen probabilistically as dispersed as possible.

As k-means++ is the extended version of k-means method, we conducted numerical experiments to evaluate and compare their performance under 1000 replicate runs. For better visual inspection and visualization, the datasets and performance values are both depicted and tabulated in Fig. 1. We can observe that k-means++ does perform better than k-means on the first three datasets. Both the clustering score (Rand Index) and time taken have been improved. However, the performance comparison is relatively complicated on the last dataset.

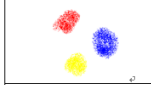
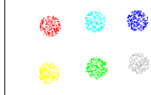
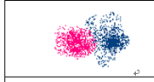
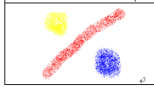
| Dataset | Methods (1000 runs) | Average Rand Index | Time taken (second) |
|---|---|---|---|
|  | k-means (k=3) | 0.98195 | 528.7969 |
| | k-means++ (k=3) | **0.99651** | **517.0938** |
|  | k-means (k=6) | 0.95778 | 127.1875 |
| | k-means++ (k=6) | **0.98689** | **105.5156** |
|  | k-means (k=2) | **0.93362** | 70.6250 |
| | k-means++ (k=2) | **0.93362** | **69.8125** |
|  | k-means (k=3) | 0.79121 | **699.0469** |
| | k-means++ (k=3) | **0.79466** | 718.2344 |

Fig. 1. Performance Comparison between k-means and k-means++.

### B. Hierarchical Clustering

Clusters are formed by following either a bottom-up approach or a top-down approach. For example, single-linkage

clustering [10] is a classic bottom-up approach in which data points are gradually agglomerated together to form clusters. In each step, all pair-wise distances are computed to identify the minimum. The parties involved in the minimal pair-wise distance are linked together. Such a step is repeated until all data points are linked together. A hierarchical tree is constructed to connect all data points at the end. A tree depth level can be chosen to cut the tree, forming clusters. To model data dynamically, a special hierarchical clustering method called Chameleon has been proposed [11]. It makes use of the inter-connectivity and closeness concept to merge and divide clusters. If the inter-connectivity and closeness between two clusters are higher than those within the clusters, then the two clusters are merged.

### C. Density-based Clustering

Apart from the well-known clustering methods, there are different clustering paradigms. In density-based clustering, data is clustered based on some connectivity and density functions. For example, DBscan [12] uses density-based notions to define clusters. Two connectivity functions *density-reachable* and *density-connected* have been proposed to define each data point as either a core point or a border point. DBscan visits points arbitrarily until all points have been visited. If the point is a core point, it tries to expand and form a cluster around itself. Based on the experimental results, the authors have demonstrated its robustness toward discovering arbitrarily shaped clusters.

### D. Grid-based Clustering

In grid-based clustering, the data space is divided into multiple portions (grids) at different granularity levels to be clustered individually. For example, CLIQUE [13] can automatically find subspaces with high density clusters. No data distribution assumption has been made. The empirical results demonstrated that it could scale well with the number of dimensions. Thus it is especially efficient in clustering high-dimensional data.

### E. Correlation Clustering

Correlation clustering [14] was motivated from a document clustering problem in which one has a pair-wise similarity function $f$ learned from past data. The goal is to partition the current set of documents in a way that correlates with $f$ as much as possible. In other words, we have a complete graph of N vertices, where each edge is labeled either $+$ or $-$. Our goal is to produce a partition of vertices (a clustering) that agrees with the edge labels. The authors have proved that this problem is a NP-complete problem. Hence they proposed two approximation algorithms to achieve the partitioning.

The first method called *Cautious* is to minimize the disagreements (number of $-$ edges inside clusters plus the number of + edges between clusters), whereas the second method called *PTAS* is to maximize the agreements (number of + edges inside clusters plus the number of $-$ edges between clusters). Basically, the ideas of the above two methods are the same (to aggregate the vertices which agree with their edge labels). The first method is discussed in detail in this work.

First, we arbitrarily choose a vertex v. Then we pick up all the positive neighbors (the neighbor vertices with $+$ edge) of the vertex and put them into a set $A$. Having picked up all the positive neighbors of the vertex, we perform pruning. That is the 'Vertex Removal Step'. In this step, we move on to check $3\delta$-bad for all the positive neighbors of the vertex, where $\delta = 1/44$. If there are, we remove it from the set $A$. After the removal step, the next step is 'Vertex Addition Step' in which we try to add back some vertices which are $7\delta$-good with the chosen vertex v to the set $A$. The vertices in the set $A$ are then chosen as one cluster. The above steps are repeated until no vertices are left or the set $A$ becomes empty.

### F. Spectral Clustering

Some of the existing clustering approaches may find local minima and require an iterative algorithm to find good clusters using different initial cluster starting points. In contrast, spectral clustering [15], [16], [17] is a relatively promising approach for clustering based on the leading eigenvectors of the matrix derived from a distance matrix. The main idea is to make use of the spectrum of the similarity matrix of the data to perform dimensionality reduction for k-means clustering in fewer dimensions. The seminal work [15] is discussed in this work.

At the beginning, we form an affinity matrix A, which is a NxN matrix and N is the total number of data points. Each entry $A_{ij}$ corresponds to the similarity measure between the data points $s_i$ and $s_j$. The scaling parameter $\sigma^2$ controls how rapidly $A_{ij}$ falls off with the distance between $s_i$ and $s_j$. After we have formed the affinity matrix A, we construct the Laplacian matrix L from the normalized affinity matrix of A. Then we find the k leading eigenvectors (i.e. with k leading eigenvalues) of L and form the matrix X by stacking the eigenvectors in column. After we have stacked the eigenvectors to form the matrix X, we normalize each row. Then we treat each row in X as a data vector and use k-means clustering algorithm to cluster them. The clustering results are projected back onto the original data (i.e. it assigns the original point $s_i$ to cluster j if and only if row i of the matrix X is assigned to cluster j).

### G. Gravitational Clustering

Distinct from the works we have mentioned, gravitational clustering is considered as a rather unique method. It was first proposed by Wright [18]. In the method, each data instance is considered as a particle within the feature space. A physical model is applied to simulate the movements of the particles. As described in [19], Jonatan et al. proposed a new gravitational clustering method using Newton laws of motion. A simplified version of gravitational clustering was proposed by Long et al. [20]. Wang et al. proposed a local shrinking method to move data toward the medians of their k nearest neighbors [21]. Blekas and Lagaris [22] proposed a similar method called Newtonian Clustering in which Newton's equations of motion are applied to shrink and separate data, followed by Gaussian mixture model building. Molecular dynamics-like mechanism was also applied for clustering by Junlin et al [23].

### H. Herd Clustering

To tackle the clustering problem, a novel clustering method, Herd Clustering (HC), has been proposed by Wong et al. [24]. It novelties lie in two aspects: (1) HC is inspired from the nature, herd behavior, which is a commonly seen phenomenon in the real world including human mobility patterns [25]. Thus it is very intuitive and easy to be understood for its good performance. (2) HC also demonstrates that cluster analysis can be done in a non-traditional way by making data *alive*.

HC differs from the traditional ones. Instead of trying hard to analyze data alone, it also spends effort on moving data. Two stages are proposed in HC.

Inspired by the herd behavior [26], an attraction model is used to guide data movements in the first stage. Each data instance is represented by a particle. The coordinate position of a particle is given by the values of the corresponding data instance it represents. The particles attract each other if their distances are smaller than a threshold. Each particle has its own velocity (initially zero). In each iteration, the velocity of a particle is affected by the neighborhood particles. If most particles are found in a particular direction, the velocity of the particle is accelerated toward that direction.

After all the iterations in the first stage, all data instances should be well separated and merged together. They are much easier to be clustered than before. Thus an intuitive approach is proposed to cluster data in the second stage. A list of cluster centroids is maintained. At the beginning, the centroid list is empty. For each point, we check whether its distance to any centroid is smaller than the threshold. If a centroid is detected, then the point is assigned the same cluster as the centroid. If its distances to all centroids are higher than or equal to the threshold, the point is added to the list and start a new cluster around it. After all data instances are scanned, a clustering result is obtained.

At the first glance, HC is similar to Gravitation Clustering (GC) [18]: data instances are moved according to a model. Nonetheless, their details are totally different. For instance, the model in GC is a physical model following Newton Laws of motion, while that in HC is an artificial model which is designed for computational efficiency. The particle acceleration decreases as the inter-particle distance increases in GC while they are independent in HC. Calculus is involved in GC whereas only computationally efficient operations are allowed in HC.

### I. Others

There are lots of other clustering methods proposed in the past. For instance, Maulik et al. applied a genetic algorithm to search for cluster centers [27]. A globally incremental approach to k-means has been reported in [28]. Celeux et al. have proposed a novel method called Gaussian parsimonious clustering models [29]. Different distance measures have been incorporated into an objective function to cluster arbitrary number of clusters [30]. A hierarchical agglomerative clustering methodology using symbolic objects has been described in [31]. Tsao et al. used a fuzzy Kohonen network for clustering [32]. A fuzzy c-means algorithm has been developed as described in [33], [34]. An alternative pruning approach to reduce the noise effect has also been proposed for the fuzzy c-means algorithm [35]. In recent years, several kernel methods have been developed for clustering [36]. A fuzzy-rough set application to microarray data has also been reported in [37]. Hu et al. have applied a hierarchical clustering method for active learning [38]. Interestingly, Corsini et al. have trained a neural network to define dissimilarity measures which are subsequently used in the relational clustering [39]. Gullo et al. have also proposed clustering methods on uncertain data [40], [41], [42]. There are many other works; more details can be found in [10], [43], [44].

## III. Advanced Clustering

### A. Clustering on Data Stream

The previous clustering methods assume data are static during clustering. Nonetheless, modern data are not static necessarily. In fact, data can be transmitted in streaming form; for instance, real-time financial stock market data, video surveillance data, and social media data. Modern data keeps itself changing and evolving during the course of clustering. For analysis of such data, the ability to process the data in a timely manner with little memory is crucial. In light of that, different data stream clustering methods are proposed. Fo instance, Guha et al. have proposed one of the first-known method, STREAM, to solve the k-median problem on streaming data with constant-factor approximation [45]. An incremental clustering method (COBWEB) has also been proposed to maintain a hierarchical clustering tree on streaming data by Fisher [46]. Zhang et al. have proposed an efficient data clustering method for large datasets [47]. Thanks to its linear complexity and single-pass nature, it can also be applied to cluster data streams with a tree data structure, CF Tree [47]. On the other hand, an incremental clustering method (C2ICM) has been proposed to data stream clustering problems. In particular, a lower bound for its clustering performance has also been provided [48].

### B. Clustering on Sequence Data

In the past years, probabilistic graphical models have been successfully applied to different problems such as gene clustering [49], [50], [51]. In particular, Hidden Markov Model (HMM) has been demonstrated successful for clustering sequence data in a wide range of domains [52].

*1) Description:* Hidden Markov Model (HMM) is a probabilistic graphical model which assumes a sequence of symbols is controlled and generated by a corresponding sequence of hidden states with the same sequence length. In particular, Markov property is assumed for the sequence of hidden states; in other words, each hidden state solely depends on its previous hidden state on the same sequence. Although such Markov assumption over-simplifies the independence between different states, it can work fairly well in practice. Moreover, it greatly reduces the computational complexity in HMM learning and

inference. Mathematically, an HMM can be described as $\theta$:

$$\theta = (\{a_{ij}\}, \{b_i(x)\}, \{\pi_i\})$$
$$\forall i, j \in \{1, 2, ..., N\}, \forall x \in X$$
$$s.t.$$
$$\sum_{i=1}^{N} \pi_i = 1$$
$$\sum_{j=1}^{N} a_{ij} = 1 \ \forall i \in \{1, 2, ..., N\}$$
$$\sum_{x \in X} b_i(x) = 1 \ \forall i \in \{1, 2, ..., N\}$$
$$0 \leq \pi_i, a_{ij}, b_i(x) \leq 1 \ \forall i, j \in \{1, 2, ..., N\}, \forall x \in X$$

where $a_{ij}$ is the transition probability from state $i$ to state $j$; $b_i(x)$ is the emission probability to emit $x$ at state $i$; $\pi_i$ is the initial state probability for state $i$.

For illustrative purposes, an HMM example with $N = 3$ hidden states is depicted in Figure 2. In that HMM example, we have 3 hidden states. At the beginning of sequence, we have the initialization probabilities $\{\pi_1, \pi_2, \pi_3\}$ for each hidden state, representing their chances to be the first hidden state. After that, the transition probabilities $\{a_{ij}\}$ determine the next hidden state recursively. For each hidden state traversal, depending on the current state, a symbol $x$ is emitted and appended to form an output sequence based on the emission probabilities $\{b_i(x)\}$.

*2) Model Learning:* To learn an HMM from sequences, Baum-Welch algorithm is usually applied to learn the unknown parameters [52]. Mathematically, Baum-Welch algorithm is an Expectation Maximization (EM) algorithm to find the maximal likelihood estimates of the HMM parameters. Thus we would like to note that Baum-Welch algorithm highly depends on the first random initialization iteration and can be trapped in local optima. Multiple runs are usually adopted to circumvent such issues. Mathematically, the Baum-Welch training algorithm can be described herein:

**Input**: A set of sequences $S = \{s_1, s_2, s_3, ..., s_M\}$ of length $L$. Each sequence $s_m$ can be represented as $s_m = s_{m1}s_{m2}...s_{mL}$ where $s_{mp} \in X \ \forall m \in \{1, 2, ..., M\}, \forall p \in \{1, 2, ..., L\}$.

**Output:** an HMM model $\theta$ trained to represent the set of sequences:

$$\theta = (\{a_{ij}\}, \{b_i(x)\}, \{\pi_i\})$$
$$\forall i, j \in \{1, 2, ..., N\}, \forall x \in X$$

where $a_{ij}$ is the transition probability from state $i$ to state $j$; $b_i(x)$ is the emission probability to emit $x$ at state $i$; $\pi_i$ is the initial state probability for state $i$.

**Method:** At the beginning of Baum-Welch algorithm, we randomly initialize those HMM model parameters $\theta_0$ and iteratively refine them in each iteration. In the expectation step (**E-step**) of the $l$-th iteration, we calculate the expected values of being in state $i$ based on the current parameter estimates $\theta_l$. Specifically, we calculate:

$$\gamma_p^m(i) = \frac{\alpha_p^m(i)\beta_p^m(i)}{P(s_m; \theta_l)}$$
$$\forall m \in \{1, 2, ..., M\}, \forall p \in \{1, 2, ..., L\}, \forall i \in \{1, 2, ..., N\}$$

where $\gamma_p^m(i)$ is the expected probability of being in state $i$ at the $p$-th position of the $m$-th sequence $s_m$; $\alpha_p^m(i)$ and $\beta_p^m(i)$ are the forward and backward probability of the $m$-th sequence $s_m$ to be in state $i$ at the $p$-th position as calculated by the dynamic programming approach [52]. $P(s_m; \theta_l)$ is the probability of observing $s_m$ given the existing HMM model parameter $\theta_l$ which can be calculated as $P(s_m; \theta_l) = \sum_{i=1}^{N} \alpha_p^m(i)\beta_p^m(i)$. In addition, we also calculate the expected values of state transitions from state $i$ to state $j$:

$$\zeta_p^m(i, j) = \frac{\alpha_p^m(i)a_{ij}b_j(s_{mp})\beta_{p+1}^m(j)}{P(s_m; \theta_l)}$$
$$\forall m \in \{1, 2, ..., M\}, \forall p \in \{1, 2, ..., L\}, \forall i, j \in \{1, 2, ..., N\}$$

where $\zeta_p^m(i, j)$ is the expected probability of transiting from state $i$ at the $p$-th position to state $j$ at the $(p+1)$-th position for the $m$-th sequence $s_m$, given the current parameter estimates $\theta_l$.

In the maximization step (**M-step**) of the $l$-th iteration, those model parameters are refined to be the maximal likelihood estimates for those expected values:

$$\pi_i' = \frac{\sum_{m=1}^{M} \gamma_1^m(i)}{M} \quad \forall i \in \{1, 2, ..., N\}$$
$$a_{ij}' = \frac{\sum_{m=1}^{M} \sum_{p=1}^{L-1} \zeta_p^m(i, j)}{\sum_{m=1}^{M} \sum_{p=1}^{L-1} \gamma_p^m(i)} \quad \forall i, j \in \{1, 2, ..., N\}$$
$$b_i(x)' = \frac{\sum_{m=1}^{M} \sum_{p=1}^{L} \gamma_p^m(i)[s_{mp} = x]}{\sum_{m=1}^{M} \sum_{p=1}^{L} \gamma_p^m(i)}$$
$$\forall i \in \{1, 2, ..., N\}, \forall x \in \{A, C, G, T, -\}$$
$$\theta_{l+1} = (\{a_{ij}'\}, \{b_i(x)'\}, \{\pi_i'\})$$
$$\forall i, j \in \{1, 2, ..., N\}, \forall x \in \{A, C, G, T, -\}$$

The new HMM model parameters $\theta_{l+1}$ are used in the next iteration. We repeat the E-step and M-step alternatively until the HMM model parameters are not changed anymore. In other words, the difference between $\theta_l$ and $\theta_{l+1}$ converges to a numerically negligible value at which a local optimum is found.

## IV. Verification

### A. Benchmark Data Sources

Benchmark datasets can be downloaded from the UCI Machine Learning Repository [53].

### B. Performance Metrics for Clustering

For clustering, Rand Index [54], Purity [55], F-measure [55], and Normalized Mutual Information (NMI) [56] are usually adopted for performance benchmarking. Rand Index is based on the intra-cluster similarity and inter-cluster dissimilarity. For the intra-cluster similarity, if a pair of data vectors is assigned the same cluster in both the target result and the clustering result, then the score will be increased by one. For

the inter-cluster dissimilarity, if a pair of vectors is assigned different clusters in both the target result and the clustering result, then the score will also be increased by one. On the contrary, if a pair of data vectors is in the same cluster in the target result, but not in the clustering result, the score will not be increased. After we have checked all the possible pairs, the score is normalized by the total number of possible pairs. Mathematically, the formula is derived as follows

$$Rand\ Index = \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} s_{ij}}{n^2 - n}\ where\ i \neq j\ ,$$

$$s_{ij} = \begin{cases} 1 & \text{if } G_o(d_i) = G_o(d_j)\ and\ G(d_i) = G(d_j). \\ 1 & \text{if } G_o(d_i) \neq G_o(d_j)\ and\ G(d_i) \neq G(d_j). \\ 0 & \text{otherwise.} \end{cases}$$

, where n is the number of data vectors, $d_i$ is the $i$th data vector, $d_j$ is the $j$th data vector, $G_o(d)$ is the cluster group id of a data vector $d$ in the target result, $G(d)$ is the cluster group id of a data vector $d$ in the clustering result. On the other hand, F-measure is similar to Rand Index with the exception that true negatives are not taken into account. Mathematically, the formula is derived as follows:

$$F - measure = \frac{2a}{2a + b + c}$$

$$\begin{aligned} a &= \sum_{i=1}^{n}\sum_{j=1}^{n}[i \neq j][G_o(d_i) = G_o(d_j)\ \&\ G(d_i) = G(d_j)]. \\ b &= \sum_{i=1}^{n}\sum_{j=1}^{n}[i \neq j][G_o(d_i) \neq G_o(d_j)\ \&\ G(d_i) = G(d_j)] \\ c &= \sum_{i=1}^{n}\sum_{j=1}^{n}[i \neq j][G_o(d_i) = G_o(d_j)\ \&\ G(d_i) \neq G(d_j)] \end{aligned}$$

, where [...] is the Iverson bracket. In contrast, purity solely measures the intra-cluster similarity. Nevertheless, it is useful in the sense that we only care about the quality of individual clusters. Mathematically, the purity of a cluster $C_i$ of size $n_i$ is defined below. For $n$ data instances with $k$ cluster groups, the overall purity of a clustering result is defined as:

$$P(C_i) = \frac{1}{n_i}\max_j(n_i^j)$$

$$Purity = \sum_{i=1}^{k}\frac{n_i}{n}P(C_i)$$

, where $n_i^j$ is the number of the data instances of $j$th class that are assigned to the $i$th cluster. To account for all the performance results, Normalized Mutual Information (NMI) can also be used [56]. For all non-deterministic methods, the performance metrics are taken by averaging over multiple runs. For all deterministic methods, the performance metrics are taken by running once only.

### C. Performance Metrics for Prediction

From the perspective of predictive tasks, a clustering outcome can be categorized into 4 types. If the clustering outcome is consistent with the truth, it is called either **True Positive (TP)** or **True Negative (TN)**, depending on the actual value. Otherwise, it is called **False Positive (FP)** or **False Negative (FN)** respectively. In different problem domains, FPs and TNs are depreciated and weighted differently. For instance, FPs are more tolerated than FNs in human disease diagnosis.

To summarize the prediction performance of a clustering method, accuracy is widely adopted. It is defined as follows:

$$Accuracy = \frac{TPs + NPs}{TPs + FNs + FPs + TNs}$$

Nonetheless, accuracy may be non-informative if the dataset is imbalanced or mis-clustering cost is very high. For instance, if only the performance of a method on positive class prediction is practically interesting, we can adopt precision and sensitivity (a.k.a. true positive rate and recall) which are defined as follows:

$$Precision = \frac{TPs}{TPs + FPs}$$

$$Sensitivity = \frac{TPs}{TPs + FNs}$$

Alternatively, F-measure can be applied to combine precision and sensitivity into a single performance metric. It is defined as the harmonic mean of precision and sensitivity. The duals of precision and sensitivity for negative class clustering are negative predictive value (NPV) and specificity respectively.

$$NPV = \frac{TNs}{TNs + FNs}$$

$$Specificity = \frac{TNs}{TNs + FPs}$$

In particular, we would like to note that the well-known false positive rate (FPR) and false discovery rate (FDR) are defined as follows:

$$FPR = 1 - Specificity\ ,\ FDR = 1 - Precision$$

Although the performance metrics described are very suitable for evaluating discrete clustering predictions. Nonetheless, the modern clustering methods usually assign a confidence value to each of its prediction. To examine the modern methods in full spectrum, receiver operating characteristics (ROC) curves and precision-recall (PRC) curves are proposed. Different thresholds are cut at the confidence values to observe the performance trade-off of each method. For instance, the trade-off between sensitivity and false positive rates can be observed from ROC curves whereas that between precision and recall can be observed from PRC curves. The area under ROC curves (AUC) is usually adopted as a benchmarking metric.

### D. Evaluation Procedures

The most typical evaluation procedure is to divide a dataset into two sets: training dataset and testing dataset. The training dataset is used for training a clustering model, while the testing dataset is isolated and reserved for testing the trained model. In particular, the most common procedure is N-fold cross-validation which has $N$ iterations. The dataset is randomly divided into N non-overlapping subsets. In each iteration, a subset is rotated as the testing dataset while the others are assigned as the corresponding training dataset. If the input data is scarce or costly, leave-one-out cross-validation can also be applied. In that case, only one data sample is left out for testing, while the others are allocated as the training dataset in each iteration.

### E. Statistical Tests

Since some of the existing clustering methods are stochastic, multiple replicate runs need to be executed for comprehensive benchmarking [24]. The means and standard deviations of performance metrics are usually reported for fair comparison. To justify the results, statistical tests are adopted to assess the statistical significances; For instance, t-tests, Mann-Whitney U-tests (MWU), and Kolmogorov-Smirnov test (KS).

## V. Benchmarking

To investigate the performance difference between those methods, four representative methods have been selected and run on different datasets. K-means++ is chosen for its simplicty and superior performance over the traditional k-means method; Correlation clustering is selected to represent the algorithms with solid theoretical support; Unsupervised optimal fuzzy clustering is chosen to represent the soft clustering algorithms; Spectral clustering is selected to represent the modern clustering algorithms. Since all of the methods selected are stochastic, 100 replicate runs are executed to compute the average performance metrics for each method on each dataset. All the parameters were tuned for each algorithm and dataset manually. The results are depicted in Fig. 3.

From the results, we can observe that the clustering methods exhibit different characteristics on different datasets. In general, based on the performance metric (Rand Index), spectral clustering is found to perform the best among the selected algorithms whereas the performance of correlation clustering is relatively limited. Based on the time taken, k-means++ is the fastest one, whereas correlation clustering is the slowest one on most datasets. The top three datasets are the most typical datasets. Each cluster forms a globular shape. It is not hard for us to expect that they can be solved by most clustering algorithms. The result turns out to concede with our expectation, except correlation clustering. The middle three datasets are difficult datasets. Each cluster is an irregular shape. Within the same dataset, each cluster is even not guaranteed to be similar to the other clusters. Interestingly, a nearly perfect result can be obtained by spectral clustering, reflecting that the dimensional transformation ability within spectral clustering does play a role in lowering the difficulties in handling such irregular data shapes. The bottom four datasets are the well-known datasets taken from the UCI machine learning repository. The number of attributes is ranged from 4 to 32. The number of class labels is ranged from 2 to 10. The number of instances is ranged from 150 to 1484. In the experiment, each algorithm has managed to perform well on a particular dataset. No conclusive insights can be drawn from the result. The data dependency of the clustering algorithms is fully reflected on those datasets.

## VI. Summary and Future Works

### A. Summary

With growing data, cluster algorithms (also known as cluster analysis) become important tools for analyzing data. In this book chapter, we have reviewed the existing clustering algorithms from different paradigms: Partitional Clustering, Hierarchical Clustering, Density-based Clustering, Grid-based Clustering, Correlation Clustering, Spectral Clustering, Gravitational Clustering, Herd Clustering, and Others. Especially, we have focused on their methodologies and design concepts. Advanced clustering methods have also been reviewed; for instance, data stream clustering and sequence clustering.

To verify the algorithms' competitiveness, different types of performance metrics have been defined and reviewed. In particular, benchmark studies have been conducted to observe the empirical performance of the selected methods: k-means++, correlation clustering, fuzzy clustering, and spectral clustering. The numerical results reveal that spectral clustering has its own competitive edge over the other methods on low-dimensional datasets. For high-dimensional datasets, we cannot observe any significant performance difference between the selected methods.

Nonetheless, during the course of the studies here, we found several future directions which we believe they are promising. They are described in the following section.

### B. Future Works

*1) Computational Scalability:* As mentioned at the very beginning of this book chapter, the recent advancements of science and technologies enable massive data generation in recent years. Some of the existing computational methods may not scale with the large amount of data. For instance, the high computational complexity of spectral clustering method [57] is no longer practical to be run on the current datasets. It is imperative for us to develop new and scalable methods to keep in pace with the data generation speed.

*2) Advanced Learning Methods:* In this book chapter, we have provided an overview on clustering. It is undeniable that other machine learning methods can be applied as well [58]; for instance, probabilistic graphical models can be developed and applied to capture/eliminate the uncertainty and noises in real world data.

*3) Domain Knowledge:* The existing clustering algorithms are built for general purposes. Domain knowledge can be incorporated if a clustering algorithm is applied to a specific task; for instance, if data is sparse, a sparse clustering algorithm can be applied to boost up the execution speed.

## References


[1] K.-C. Wong and Z. Zhang, "Snpdryad: predicting deleterious non-synonymous human snps using only orthologous protein sequences," *Bioinformatics*, p. btt769, Jan 2014.

[2] K.-C. Wong, K.-S. Leung, and M.-H. Wong, "Effect of spatial locality on an evolutionary algorithm for multimodal optimization," in *Proceedings of the 2010 international conference on Applications of Evolutionary Computation - Volume Part I*, ser. EvoApplicatons'10. Berlin, Heidelberg: Springer-Verlag, 2010, pp. 481–490. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-12239-2_50

[3] K.-C. Wong, C. Peng, M.-H. Wong, and K.-S. Leung, "Generalizing and learning protein-dna binding sequence representations by an evolutionary algorithm," *Soft Comput.*, vol. 15, no. 8, pp. 1631–1642, Aug. 2011. [Online]. Available: http://dx.doi.org/10.1007/s00500-011-0692-5

[4] J. de Maillard, "Un monde sans loi : La criminalit financire en images," *Editions Stock*, p. 28, 1999.

[5] T. Gonzalez, "On the computational complexity of clustering and related problems," in *System Modeling and Optimization*, ser. Lecture Notes in Control and Information Sciences, R. Drenick and F. Kozin, Eds. Springer Berlin / Heidelberg, 1982, vol. 38, pp. 174–182, 10.1007/BFb0006133. [Online]. Available: http://dx.doi.org/10.1007/BFb0006133

[6] H. Steinhaus, "Sur la division des corps ma4eriels en parties." *Bull. Acad. Pol. Sci., Cl. III*, vol. 4, pp. 801–804, 1957.

[7] G. Stockman and L. G. Shapiro, *Computer Vision*, 1st ed. Upper Saddle River, NJ, USA: Prentice Hall PTR, 2001.

[8] H. Xiong, J. Wu, and J. Chen, "K-means clustering versus validation measures: A data-distribution perspective," *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, vol. 39, no. 2, pp. 318–331, 2009.

[9] D. Arthur and S. Vassilvitskii, "k-means++: the advantages of careful seeding," in *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, ser. SODA '07. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, 2007, pp. 1027–1035. [Online]. Available: http://portal.acm.org/citation.cfm?id=1283383.1283494

[10] R. Xu and D. Wunsch, "Survey of Clustering Algorithms," *IEEE Transactions on Neural Networks*, vol. 16, no. 3, pp. 645–678, May 2005. [Online]. Available: http://dx.doi.org/10.1109/TNN.2005.845141

[11] G. Karypis, E.-H. S. Han, and V. Kumar, "Chameleon: Hierarchical clustering using dynamic modeling," *Computer*, vol. 32, pp. 68–75, August 1999. [Online]. Available: http://dx.doi.org/10.1109/2.781637

[12] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proc. of 2nd International Conference on Knowledge Discovery and Data Mining*, 1996, pp. 226–231.

[13] R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan, "Automatic subspace clustering of high dimensional data for data mining applications," *SIGMOD Rec.*, vol. 27, pp. 94–105, June 1998. [Online]. Available: http://doi.acm.org/10.1145/276305.276314

[14] N. Bansal, A. Blum, and S. Chawla, "Correlation clustering," *Machine Learning Journal*, vol. Special Issue on Theoretical Advances in Data Clustering, pp. 86–113, 2004.

[15] A. Y. Ng, M. I. Jordan, and Y. Weiss, "On spectral clustering: Analysis and an algorithm," in *Advances in Neural Information Processing Systems*. MIT Press, 2001, pp. 849–856.

[16] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, pp. 888–905, August 2000. [Online]. Available: http://dx.doi.org/10.1109/34.868688

[17] M. Maila and J. Shi, "A random walks view of spectral segmentation," in *AI and STATISTICS (AISTATS) 2001*, 2001.

[18] W. Wright, "Gravitational clustering," *Pattern Recognition*, vol. 9, no. 3, pp. 151 – 166, 1977. [Online]. Available: http://www.sciencedirect.com/science/article/pii/0031320377900139

[19] J. Gomez, D. Dasgupta, and O. Nasraoui, "A new gravitational clustering algorithm," in *In Proc. of the SIAM Int. Conf. on Data Mining (SDM*, 2003.

[20] T. Long and L.-W. Jin, "A new simplified gravitational clustering method for multi-prototype learning based on minimum classification error training," in *Advances in Machine Vision, Image Processing, and Pattern Analysis*, ser. Lecture Notes in Computer Science, N. Zheng, X. Jiang, and X. Lan, Eds. Springer Berlin / Heidelberg, 2006, vol. 4153, pp. 168–175.

[21] X. Wang, W. Qiu, and R. H. Zamar, "Clues: A non-parametric clustering method based on local shrinking," *Computational Statistics & Data Analysis*, vol. 52, no. 1, pp. 286–298, September 2007. [Online]. Available: http://ideas.repec.org/a/eee/csdana/v52y2007i1p286-298.html

[22] K. Blekas and I. E. Lagaris, "Newtonian clustering: An approach based on molecular dynamics and global optimization," *Pattern Recogn.*, vol. 40, pp. 1734–1744, June 2007. [Online]. Available: http://portal.acm.org/citation.cfm?id=1231536.1231754

[23] L. Junlin and F. Hongguang, "Molecular dynamics-like data clustering approach," *Pattern Recognition*, vol. 44, no. 8, pp. 1721 – 1737, 2011. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0031320311000173

[24] K.-C. Wong, C. Peng, Y. Li, and T.-M. Chan, "Herd clustering: A synergistic data clustering approach using collective intelligence," *Applied Soft Computing*, vol. 23, pp. 61–75, 2014.

[25] C. Peng, X. Jin, K.-C. Wong, M. Shi, and P. Liò, "Collective human mobility pattern from taxi trips in urban area," *PloS one*, vol. 7, no. 4, p. e34487, 2012.

[26] A. V. Banerjee, "A Simple Model of Herd Behavior," *The Quarterly Journal of Economics*, vol. 107, no. 3, pp. 797–817, August 1992. [Online]. Available: http://dx.doi.org/10.2307/2118364

[27] U. Maulik and S. Bandyopadhyay, "Genetic algorithm-based clustering technique," *Pattern Recognition*, vol. 33, no. 9, pp. 1455 – 1465, 2000. [Online]. Available: http://www.sciencedirect.com/science/article/B6V14-40961WK-5/2/41f336383004b7f397ae8e9266f90b0a

[28] A. Likas, N. Vlassis, and J. J. Verbeek, "The global k-means clustering algorithm," *Pattern Recognition*, vol. 36, no. 2, pp. 451 – 461, 2003. [Online]. Available: http://www.sciencedirect.com/science/article/B6V14-45TTJY0-7/2/1bfdf4def0cf8b6ba77fd25132df8d0d

[29] G. Celeux and G. Govaert, "Gaussian parsimonious clustering models," *Pattern Recognition*, vol. 28, no. 5, pp. 781 – 793, 1995. [Online]. Available: http://www.sciencedirect.com/science/article/B6V14-3YGV1N5-16/2/2f50595a70e52e39054440b62548439e

[30] H. Frigui and R. Krishnapuram, "Clustering by competitive agglomeration," *Pattern Recognition*, vol. 30, no. 7, pp. 1109 – 1119, 1997. [Online]. Available: http://www.sciencedirect.com/science/article/B6V14-3SNVHWM-M/2/3c5ec045f0a0662d00ab583aab028f9f

[31] K. C. Gowda and E. Diday, "Symbolic clustering using a new dissimilarity measure," *Pattern Recognition*, vol. 24, no. 6, pp. 567 – 578, 1991. [Online]. Available: http://www.sciencedirect.com/science/article/B6V14-48MPNTW-1JS/2/7ae7f4c958c1acf937f4f02091c5d073

[32] E. C.-K. Tsao, J. C. Bezdek, and N. R. Pal, "Fuzzy kohonen clustering networks," *Pattern Recognition*, vol. 27, no. 5, pp. 757 – 764, 1994. [Online]. Available: http://www.sciencedirect.com/science/article/B6V14-48MPPXD-1YX/2/cf0e26b7498b3a4f4b7698975495f7f9

[33] K.-L. Wu and M.-S. Yang, "Alternative c-means clustering algorithms," *Pattern Recognition*, vol. 35, no. 10, pp. 2267 – 2278, 2002. [Online]. Available: http://www.sciencedirect.com/science/article/B6V14-44JD474-1/2/790f559ea2639362d45159d141bfd583

[34] L. Zhu, F.-L. Chung, and S. Wang, "Generalized fuzzy c-means clustering algorithm with improved fuzzy partitions," *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, vol. 39, pp. 578–591, June 2009. [Online]. Available: http://dl.acm.org/citation.cfm?id=1656753.1656754

[35] J.-S. Zhang and Y.-W. Leung, "Robust clustering by pruning outliers." *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, vol. 33, no. 6, pp. 983–998, 2003. [Online]. Available: http://dblp.uni-trier.de/db/journals/tsmc/tsmcb33.html#ZhangL03

[36] C. F. M. F. R. S. Filippone, M., "A survey of kernel and spectral methods for clustering," *Pattern Recognition*, vol. 41, no. 1, pp. 176–190, 2008. [Online]. Available: http://www.scopus.com/inward/record.url?eid=2-s2.0-34548025132&partnerID=40&md5=2d78a0f28c666bd207ce98688b66ca9b

[37] P. Maji, "Fuzzy-rough supervised attribute clustering algorithm and classification of microarray data," *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, vol. 41, no. 1, pp. 222–233, 2011.

[38] W. Hu, W. Hu, N. Xie, and S. Maybank, "Unsupervised active learning based on hierarchical graph-theoretic clustering," *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, vol. 39, pp. 1147–1161, October 2009. [Online]. Available: http://dl.acm.org/citation.cfm?id=1656796.1656802

[39] P. Corsini, B. Lazzerini, and F. Marcelloni, "A fuzzy relational clustering algorithm based on a dissimilarity measure extracted from data," *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, vol. 34, no. 1, pp. 775 – 781, feb. 2004.

[40] F. Gullo, G. Ponti, and A. Tagarelli, "Clustering uncertain data via k-medoids," in *Proceedings of the 2Nd International Conference on Scalable Uncertainty Management*, ser. SUM '08. Berlin, Heidelberg: Springer-Verlag, 2008, pp. 229–242. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-87993-0_19

[41] F. Gullo, G. Ponti, A. Tagarelli, and S. Greco, "A hierarchical algorithm for clustering uncertain data via an information-theoretic approach," in

*Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*. IEEE, 2008, pp. 821–826.

[42] F. Gullo and A. Tagarelli, "Uncertain centroid based partitional clustering of uncertain data," *Proceedings of the VLDB Endowment*, vol. 5, no. 7, pp. 610–621, 2012.

[43] A. Baraldi and P. Blonda, "A survey of fuzzy clustering algorithms for pattern recognition. i," *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, vol. 29, no. 6, pp. 778–785, 1999.

[44] K.-C. Wong, C.-H. Wu, R. K. P. Mok, C. Peng, and Z. Zhang, "Evolutionary multimodal optimization using the principle of locality," *Inf. Sci.*, vol. 194, pp. 138–170, Jul. 2012. [Online]. Available: http://dx.doi.org/10.1016/j.ins.2011.12.016

[45] S. Guha, A. Meyerson, N. Mishra, R. Motwani, and L. O'Callaghan, "Clustering data streams: Theory and practice," *IEEE Trans. on Knowl. and Data Eng.*, vol. 15, no. 3, pp. 515–528, Mar. 2003. [Online]. Available: http://dx.doi.org/10.1109/TKDE.2003.1198387

[46] D. Fisher, "Optimization and simplification of hierarchical clusterings." in *KDD*, 1995, pp. 118–123.

[47] T. Zhang, R. Ramakrishnan, and M. Livny, "Birch: An efficient data clustering method for very large databases," in *Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD '96. New York, NY, USA: ACM, 1996, pp. 103–114. [Online]. Available: http://doi.acm.org/10.1145/233269.233324

[48] M. Charikar, C. Chekuri, T. Feder, and R. Motwani, "Incremental clustering and dynamic information retrieval," in *Proceedings of the twenty-ninth annual ACM symposium on Theory of computing*. ACM, 1997, pp. 626–635.

[49] B. J. Frey, N. Mohammad, Q. D. Morris, W. Zhang, M. D. Robinson, S. Mnaimneh, R. Chang, Q. Pan, E. Sat, J. Rossant, B. G. Bruneau, J. E. Aubin, B. J. Blencowe, and T. R. Hughes, "Genome-wide analysis of mouse transcripts using exon microarrays and factor graphs," *Nat. Genet.*, vol. 37, no. 9, pp. 991–996, Sep 2005.

[50] B. J. Frey and D. Dueck, "Clustering by passing messages between data points," *Science*, vol. 315, no. 5814, pp. 972–976, Feb 2007.

[51] Y. Barash, J. A. Calarco, W. Gao, Q. Pan, X. Wang, O. Shai, B. J. Blencowe, and B. J. Frey, "Deciphering the splicing code," *Nature*, vol. 465, no. 7294, pp. 53–59, May 2010.

[52] L. R. Rabiner, "Readings in speech recognition," A. Waibel and K.-F. Lee, Eds. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1990, ch. A tutorial on hidden Markov models and selected applications in speech recognition, pp. 267–296. [Online]. Available: http://dl.acm.org/citation.cfm?id=108235.108253

[53] A. Frank and A. Asuncion, "UCI machine learning repository," 2010. [Online]. Available: http://archive.ics.uci.edu/ml

[54] M. Halkidi, Y. Batistakis, and M. Vazirgiannis, "On clustering validation techniques," *J. Intell. Inf. Syst.*, vol. 17, pp. 107–145, December 2001. [Online]. Available: http://portal.acm.org/citation.cfm?id=607585.607609

[55] Y. Zhao and G. Karypis, "Criterion functions for document clustering: Experiments and analysis," University of Minnesota, Department of Computer Science, Tech. Rep., 2002.

[56] N. X. Vinh, J. Epps, and J. Bailey, "Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance," *J. Mach. Learn. Res.*, vol. 11, pp. 2837–2854, Dec. 2010. [Online]. Available: http://dl.acm.org/citation.cfm?id=1756006.1953024

[57] T. L. Bailey and C. Elkan, "The value of prior knowledge in discovering motifs with MEME," *Proc Int Conf Intell Syst Mol Biol*, vol. 3, pp. 21–29, 1995.

[58] K.-C. Wong, Y. Li, C. Peng, and Z. Zhang, "Signalspider: probabilistic pattern discovery on multiple normalized chip-seq signal profiles," *Bioinformatics*, p. btu604, 2014.

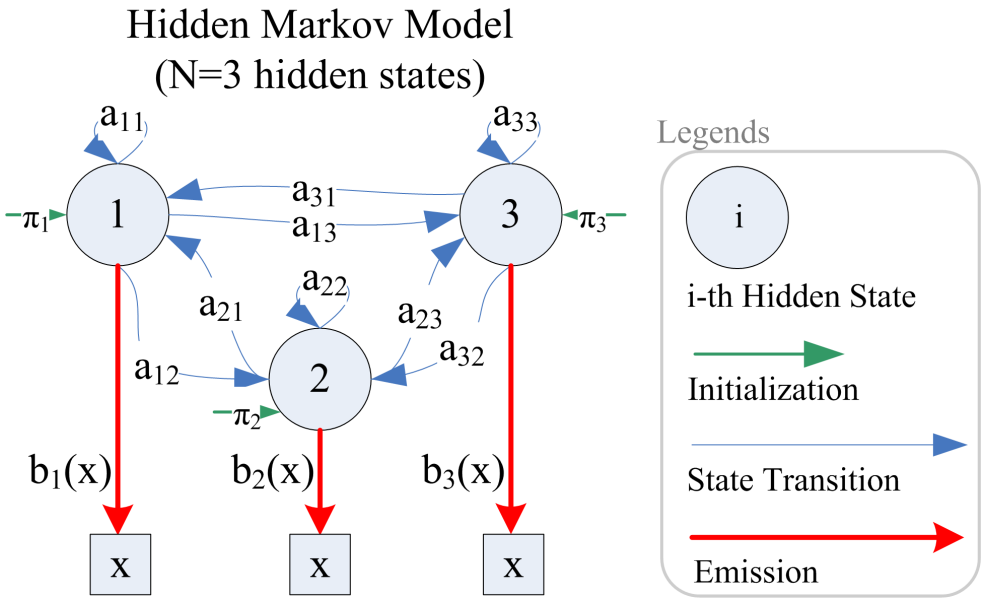



Fig. 2. Hidden Markov Model (HMM) example with $N = 3$ hidden states.

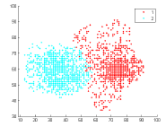
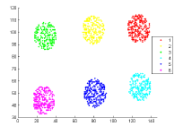
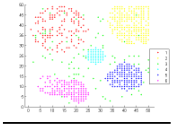
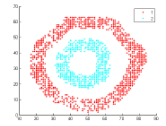
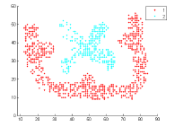
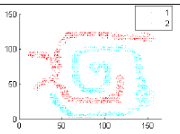
| | Dataset | 100 runs | k-means++ | Correlation Clustering | Unsupervised Optimal Fuzzy Clustering | Spectral Clustering |
|---|---|---|---|---|---|---|
| 1 |  | Setting | K=2 | T=40 | K=2, stoping threshold = 0.1, g = 0.2 | K=2, $\sigma^2$=1 |
| | | Rand Index | 0.93362 | 0.50482 | 0.93593 | 0.84499 |
| | | Time (sec) | 7.1 | 28427 | 22.3 | 2240.4 |
| 2 |  | Setting | K=6 | T=30 | K=6, stoping threshold = 0.1, g = 0.2 | K=6, $\sigma^2$=1 |
| | | Rand Index | 0.99291 | 0.79183 | 0.99312 | 0.9479 |
| | | Time (sec) | 10.1 | 1053.9 | 101.6 | 3788.7 |
| 3 |  | Setting | K=5 | T=20 | K=5, stoping threshold = 0.1, g = 15 | K=5, $\sigma^2$=20 |
| | | Rand Index | 0.93335 | 0.77976 | 0.92286 | 0.94192 |
| | | Time (sec) | 9.9 | 1174.3 | 51.2500 | 606 |
| 4 |  | Setting | K=2 | T=50 | K=2, stoping threshold = 0.1, g = 0.2 | K=2, $\sigma^2$=1 |
| | | Rand Index | 0.50147 | 0.52527 | 0.50197 | 1 |
| | | Time (sec) | 14.8 | 18642 | 57.6 | 2955.2 |
| 5 |  | Setting | K=2 | T=43 | K=2, stoping threshold = 0.1, g = 0.2 | K=2, $\sigma^2$=1 |
| | | Rand Index | 0.49968 | 0.53187 | 0.50091 | 1 |
| | | Time (sec) | 6.2 | 10141 | 20.4 | 929.1 |
| 6 |  | Setting | K=2 | T=80 | K=2, stoping threshold = 0.1, g = 0.2 | K=2, $\sigma^2$=1 |
| | | Rand Index | 0.51742 | 0.50304 | 0.51805 | 1 |
| | | Time (sec) | 19.4 | 66962 | 151.9 | 9489.0 |
| 7 | **Iris dataset**<br>4 attributes<br>1 class label ( 3 classes)<br>150 instances<br>http://archive.ics.uci.edu/ml/datasets/Iris | Setting | K=3 | T=2 | K=3, stoping threshold = 0.1, g = 0.2 | K=3, $\sigma^2$=1 |
| | | Rand Index | 0.86536 | 0.82284 | 0.86606 | 0.87192 |
| | | Time (sec) | 0.7813 | 58.8594 | 2.1875 | 13.6094 |
| 8 | **Wine dataset**<br>12 attributes<br>1 class label ( 3 classes)<br>178 instances<br>http://archive.ics.uci.edu/ml/datasets/Wine | Setting | K=3 | T=25 | K=3, stoping threshold = 0.1, g = 0.1 | K=3, $\sigma^2$=20 |
| | | Rand Index | 0.70915 | 0.66979 | 0.71035 | 0.68176 |
| | | Time (sec) | 1.1719 | 126.5781 | 17.2188 | 22.6719 |
| 9 | **Yeast dataset**<br>8 attributes<br>1 class label (10 classes)<br>1484 instances<br>http://archive.ics.uci.edu/ml/datasets/Yeast | Setting | K=10 | T=0.2 | K=10, stoping threshold = 0.1, g = 0.2 | K=10, $\sigma^2$=20 |
| | | Rand Index | 0.74245 | 0.7616 | 0.72487 | 0.75653 |
| | | Time (sec) | 99.1875 | 8248.6 | 32.3281 | 3631.5 |
| 10 | **Breast Cancer Wisconsin (Diagnosis) dataset**<br>32 attributes<br>1 class label ( 2 classes)<br>569 instances<br>http://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+%28Diagnostic%29 | Setting | K=2 | T=1000 | K=2, stoping threshold = 0.1, g = 10 | K=2, $\sigma^2$=20 |
| | | Rand Index | 0.75038 | 0.50586 | 0.75038 | 0.7405 |
| | | Time (sec) | 2.6875 | 1399.0 | 63.6094 | 238.3906 |

Fig. 3. Performance Comparison between the methods selected.