Adaptive Mean Shift-Based Clustering

Fajie Li[1] and Reinhard Klette[2]

[1] Institute for Mathematics and Computing Science, University of Groningen
P.O. Box 800, 9700 AV Groningen, The Netherlands
[2] Computer Science Department, The University of Auckland
Private Bag 92019, Auckland 1142, New Zealand

Abstract. This report proposes an adaptive mean shift clustering algorithm. Its application is demonstrated for simulated data, by finding old clusters which are highly overlapping. The obtained clustering result is actually close to an estimated upper bound, derived for those simulated data elsewhere.

Key words: clustering, mean shift based-clustering, recovery rate

1 Introduction

The term *cluster analysis* was coined in [6] in 1954. There are books about clustering, such as [2, 10, 16, 31], published in the 1970*s*, [20, 29, 32] published in the 1980*s*, and [3, 12] from the 1990*s*. Cluster analysis was also surveyed or reviewed in a number of papers; see, for example, [9, 13, 19, 21, 22]. For a comparisons of various clustering algorithms, see [1, 25, 28]. Examples of recent (after the year 2000) publications or PhD theses on clustering are [4, 5, 7, 8, 14, 24, 33–36]. Altogether, there are hundreds of clustering algorithms proposed in the literature; for lists of examples, see page 5 in [22], page 13 in [24] or page 130 in [30]. This all illustrates that clustering problems are very important, and often also difficult to solve.

Clustering not only interests scientists in computing but also, for example, astronomers [11, 17, 23]: Given is an observed set of stars (considered to be a set of points); how to find (*recover*) clusters which are the contributing galaxies to the observed union of those clusters?

This report discusses clustering at a general level. We illustrate by using one (fairly complex) data set of simulated astronomical data. [3] Clusters in those data are typically characterized by being highly overlapping. Obviously, the recovery of highly overlapping data is difficult, if not even (nearly) impossible.

Currently published cluster algorithms (see, for example, [34]) work neither efficiently nor correctly on such data, and upper bounds for recovery rates would allow to identify realistic goals. See [26] for the definition of recovery rate as used in this report, and [27] for the estimation of a best possible upper bound for this

[3] See `www.astro.rug.nl/~ahelmi/simulations_gaia.tar.gz` and Section 4.1 in [27].

recovery rate, illustrated there for the same simulated astronomical data example as also used in this report.

This report proposes a variant of adaptive mean shift clustering, and we illustrate it for finding old clusters in the mentioned simulated data. We compute the recovery rate of our proposed algorithm; we show that this is close to the estimated lowest possible upper bound in [27] of recovery rates for the used example of simulated data.

The rest of this report is organized as follows: Section 2 reviews an adaptive mean shift procedure and a locality-sensitive hashing method. Section 3 reviews a fast adaptive mean shift algorithm and describes our main algorithm which is applied to the specified example of simulated astronomical data, with experimental results in Section 4. Section 5 concludes the report.

2 Preliminaries

This section reviews the kernel density estimator and adaptive mean shift procedure. The first is a mathematical basis of the latter, which is the basic principle of adaptive mean shift-based clustering.

2.1 The Kernel Density Estimator

The basic idea of a kernel density estimator is a mathematical principle, implemented by Procedure 1 in Subsection 2.2. To prepare for this, we recall some definitions from [30]. We consider the kernel estimator in $\mathbb{R}^1$. Let X be a random variable. Let X_i, where $i = 1, 2, \ldots, n$, be n observations of X. We want to estimate the density of X. Let f be the density of X, and $\hat{f}$ an estimator of f. A *bin* is an interval $[a, b)$. $b-a$ is called the width of bin $[a, b)$. Let $O = (0, 0)$ be the origin, and $[mh, (m+1)h)$ are some bins of constant width h, for integers m. The histogram is defined as follows,

$$\hat{f}(x) = \frac{1}{nh} \times n_i$$

where n_i is the number of $X_i s$ in the same bin as x.

The histogram is the oldest and most frequently used density estimator. But it comes with drawbacks. For example, the histogram is not continuous (thus, it does not have derivatives); it depends on the position of the origin; and a technique based on histograms is difficult to generalize to the multivariate case.

The *naive estimator* is a generalization of the histogram technique. Since f is the density function of X, we have

$$f(x) = \lim_{h \to 0} \frac{1}{2h} \times P(X \in (x-h, x+h)) \tag{1}$$

where $P(X \in (x-h, x+h))$ is the probability of X falling into $(x-h, x+h)$. The naive estimator is defined as follows:

$$\hat{f}(x) = \frac{1}{2nh} \times n_i' \tag{2}$$

where n_i' is the number of $X_i s$ falling into $(x-h, x+h)$.

If x is the center of a bin of the current histogram, then Equation (2) will be equal to Equation (1). Therefore, the naive estimator is a generalization of the histogram technique. Equation (2) can also be written as

$$\hat{f}(x) = \frac{1}{nh} \sum_{i=1}^{n} w(\frac{x - X_i}{h}) \tag{3}$$

where w is defined as

$$w(x) = \begin{cases} 1/2 & \text{if } x \in (-1, 1) \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

The naive estimator "inherits" some drawbacks of the histogram technique. For example, it is not continuous: it "jumps" at points $X_i \pm h$ and has a zero derivative for each $x \neq X_i \pm h$.

The naive estimator can be further generalized to a *kernel estimator*: Replace function w in Equation (3) by a function K; we obtain

$$\hat{f}(x) = \frac{1}{nh} \sum_{i=1}^{n} K(\frac{x - X_i}{h}) \tag{5}$$

where K is requested to satisfy the following condition

$$\int_{-\infty}^{\infty} K(x)dx = 1 \tag{6}$$

because $K(x)$ is a density function.

If K has a derivative, for each $x \in (-\infty, +\infty)$, then [by Equation (5)] so does the kernel estimator. For example, let K be the normal density function, to construct a kernel estimator which has a derivative for each $x \in (-\infty, +\infty)$. In Equation (5), h is called the *window width (smoothing parameter,* or *bandwidth)*; K is called the *kernel* function.

Analogously, we can define the kernel estimator in $\mathbb{R}^d$:

$$\hat{f}(\mathbf{x}) = \frac{1}{nh^d} \sum_{i=1}^{n} K(\frac{\mathbf{x} - \mathbf{X_i}}{h}) \tag{7}$$

where K satisfies the following condition

$$\int_{R^d} K(\mathbf{x})d\mathbf{x} = 1 \tag{8}$$

In summary, the kernel estimator is an estimator of the density function of a random variable, represented by a sum of some simple kernel functions such that the estimator has "good" mathematical properties such as being differentiable, symmetric, and so forth.

2.2 Adaptive Mean Shift Procedure

This subsection explains the principle of adaptive mean shift-based clustering, meaning that the data points always move to local density maxima.

We review some results of [8]. Let $k(x)$ be a symmetric univariate kernel function, where $x \in (-\infty, +\infty)$. We can construct a dD kernel function from $k(x)$ as follows:

$$K(\mathbf{x}) = c_{k,d} k(\|\mathbf{x}\|^2) \tag{9}$$

where

$$c_{k,d} = \frac{\int_{R^d} K(\mathbf{x}) d\mathbf{x}}{\int_{R^d} k(\|\mathbf{x}\|^2) d\mathbf{x}} = \frac{1}{\int_{R^d} k(\|\mathbf{x}\|^2) d\mathbf{x}} > 0$$

Function $k(x)$, where $x \in [0, +\infty)$, is called the *profile* of kernel $K(\mathbf{x})$.

By Equation (9), the kernel density estimator [see Equation (7)] can be rewritten as follows:

$$\hat{f}_{h,K}(\mathbf{x}) = \frac{c_{k,d}}{nh^d} \sum_{i=1}^{n} k(\| \frac{\mathbf{x} - \mathbf{X_i}}{h} \|^2) \tag{10}$$

Suppose that $k(x)$ is differentiable in $[0, +\infty)$, except for a finite number of points. Let

$$g(x) = -k'(x)$$

where $x \in [0, +\infty)$, except for a finite number of points.

Construct a kernel function from $g(x)$ as follows:

$$G(\mathbf{x}) = c_{g,d} g(\|\mathbf{x}\|^2)$$

where

$$c_{g,d} = \frac{\int_{R^d} G(\mathbf{x}) d\mathbf{x}}{\int_{R^d} g(\|\mathbf{x}\|^2) d\mathbf{x}} = \frac{1}{\int_{R^d} g(\|\mathbf{x}\|^2) d\mathbf{x}} > 0$$

Denote the gradient of the density estimator of $\hat{f}_{h,K}(\mathbf{x})$ by $\bigtriangledown \hat{f}_{h,K}(\mathbf{x})$. Furthermore [see [8] for the details], let

$$\mathbf{m}_{h,G}(\mathbf{x}) = \frac{1}{2} h^2 \frac{c_{g,d}}{c_{k,d}} \times \frac{\bigtriangledown \hat{f}_{h,K}(\mathbf{x})}{\hat{f}_{h,G}(\mathbf{x})} \tag{11}$$

where

$$\mathbf{m}_{h,G}(\mathbf{x}) = \frac{\sum_{i=1}^{n} \mathbf{x_i} g(\| \frac{\mathbf{x} - \mathbf{X_i}}{h} \|^2)}{\sum_{i=1}^{n} g(\| \frac{\mathbf{x} - \mathbf{X_i}}{h} \|^2)} - \mathbf{x} \tag{12}$$

Equation (12) is the difference between the weighted mean and $\mathbf{x}$, known as *mean shift vector*. Since the gradient of the density estimator always points towards that direction in which the density rises most quickly, by Equation (12), the mean shift vector always points towards the direction in which the density rises most quickly. This is the main principle of mean shift-based clustering.

[7] generalized Equation (7) such that each data point $\mathbf{x_i}$ is associated with a bandwidth value h_i, where $i = 1, 2, \ldots, n$:

$$\hat{f}_K(\mathbf{x}) = \frac{c_{k,d}}{n} \sum_{i=1}^{n} \frac{1}{h_i^d} k(\| \frac{\mathbf{x} - \mathbf{X_i}}{h_i} \|^2) \tag{13}$$

Consequently, Equation (12) was generalized to

$$\mathbf{m}_G(\mathbf{x}) = \frac{\sum_{i=1}^{n} \frac{1}{h_i^{d+2}} \mathbf{x_i} g(\| \frac{\mathbf{x}-\mathbf{X_i}}{h_i} \|^2)}{\sum_{i=1}^{n} \frac{1}{h_i^{d+2}} g(\| \frac{\mathbf{x}-\mathbf{X_i}}{h_i} \|^2)} - \mathbf{x} \tag{14}$$

which is called the *adaptive* mean shift vector. Analogously, the adaptive mean shift vector also always points towards the direction in which the density rises most quickly, which is called the *mean shift property*. This is the basic principle of adaptive mean shift-based clustering. – $\| \cdot \|_1$ is the L_1 norm.

Procedure 1 *Adaptive Mean Shift Procedure*
Input: *A positive integer k, n data points* $\mathbf{x_i}$*, where i = 1, 2, ..., n, one of the data points,* $\mathbf{y_1}$*, and a total number N of iterations.*
Output: *An approximate local maximum (*mode*) of the density.*

1: $h_i = \|\mathbf{x_i} - \mathbf{x_{i,k}}\|_1$, where $\mathbf{x_{i,k}}$ is the k-nearest neighbor of $\mathbf{x_i}$
2: $j = 1$
3: **while** $j < N$ **do**
4:

$$\mathbf{y_2} = \frac{\sum_{i=1}^{n} \frac{1}{h_i^{d+2}} \mathbf{x_i} g(\| \frac{\mathbf{y_1}-\mathbf{X_i}}{h_i} \|^2)}{\sum_{i=1}^{n} \frac{1}{h_i^{d+2}} g(\| \frac{\mathbf{y_1}-\mathbf{X_i}}{h_i} \|^2)} \tag{15}$$

5: $\mathbf{y_1} = \mathbf{y_2}$
6: $j = j + 1$
7: **end while**
8: Output $\mathbf{y_2}$

If h_i is fixed in Procedure 1, then this is called the *Mean Shift Procedure*, implementing the main principle of mean shift-based clustering.

2.3 Locality-Sensitive Hashing

The main computation of the adaptive mean shift procedure is in Step 4 [Equation (15)] which has time complexity $\mathcal{O}(n^2 dN)$, where n is the number of input points, d the dimension of the input points, and N the number of iterations [14]. An efficient approximate computation of Step 4 can be achieved by solving the problem of nearest neighbor search [15]: Consider points $\mathbf{x_i}$ which are nearest to $\mathbf{y_1}$ to test whether the kernel of $\mathbf{x_i}$ "covers" $\mathbf{y_1}$. In other words, we have that $g(\| \frac{\mathbf{y_1}-\mathbf{X_i}}{h_i} \|^2) \neq 0$ in Equation (15). Also, by Equations (3) and (4), we can

see that, the larger the distance between point x and $\mathbf{X_i}$, the more likely that the value of $w(x)$ equals 0. [14] applied an approximate nearest neighbor search technique called *locality-sensitive hashing* (LSH) (see [15] and [18]) to efficiently improve the neighborhood queries.

The main ideas of LSH are represented by the following two procedures. Procedure 2 is used to partition the given data points into $K/d+1$ regions such that the points in each region have the same KD boolean vector. It is applied in Steps 2 and 5 of the Locality-Sensitive Hashing procedure (Procedure 3).

Procedure 2 *KD(i.e., K-dimensional) Boolean Vector*
Input: *A positive integer K, n data points* $\mathbf{x_i}$*, where i = 1, 2, ..., n, and a point* $\mathbf{q} = (q_1, q_2, \ldots, q_n)$.
Output: *A KD boolean vector, denoted by* $(b_1, b_2, \ldots, b_K)$.

1: $i = 1$
2: Randomly take an integer $d_k \in \{1, 2, \ldots, d\}$
3: Randomly take a real number $v_k \in [min_{x_{d_k}}, max_{x_{d_k}}]$, where

$$min_{x_{d_k}} = \min\{x_{i_{d_k}} : x_{i_{d_k}} \text{is } d_k - th \text{ coordinate of } \mathbf{x_i} \wedge i \in \{1, 2, \ldots, n\}\}$$

$$max_{x_{d_k}} = \max\{x_{i_{d_k}} : x_{i_{d_k}} \text{is } d_k - th \text{ coordinate of } \mathbf{x_i} \wedge i \in \{1, 2, \ldots, n\}\}$$

4: **if** $q_{d_k} \leq v_k$ **then**
5: $b_i = \text{true}$
6: **else**
7: $b_i = \text{false}$
8: **end if**
9: **if** $i < K$ **then**
10: $i = i + 1$ and goto Step 1
11: **end if**
12: Output $(b_1, b_2, \ldots, b_K)$

Procedure 3 *Locality-Sensitive Hashing*
Input: *Two positive integers K and L, n data points* $\mathbf{x_i}$*, where i = 1, 2, ..., n, and a query point* $\mathbf{q} = (q_1, q_2, \ldots, q_n)$.
Output: *Union (i.e., set) of points in an approximate neighborhood of* $\mathbf{q}$.

1: **for** $l \in \{1, 2, \ldots, L\}$ **do**
2: Apply Procedure 2 to compute a KD vector for $\mathbf{q}$, denoted by $\mathbf{v_q}$.
3: $C_l = \emptyset$
4: **for** $i \in \{1, 2, \ldots, n\}$ **do**
5: Apply Procedure 2 to compute a KD vector for $\mathbf{x_i}$, denoted by $\mathbf{v_{x_i}}$
6: **if** $\mathbf{v_q} = \mathbf{v_{x_i}}$ **then**
7: $C_l = C_l \cup \{\mathbf{x_i}\}$
8: **end if**
9: **end for**
10: **end for**

11: Output $\cup_{l=1}^{L} C_l$

C_l in this Procedure 3 may be implemented by a hash table. [4] – For a given K, the larger the value of L, the larger the set of points in an approximate neighborhood of $\mathbf{q}$. Thus, for a given radius h, there exists an optimal L such that Procedure 3 outputs an optimal approximate neighborhood of $\mathbf{q}$ with radius h. If L is larger than the optimal value, Procedure 3 will run for more time but will not improve the output. [14] presented a learning method to select optimal values of K and L.

3 Algorithm

Based on a locality-sensitive hashing technique, [14] derived a fast adaptive mean shift-based clustering algorithm (Algorithm 1) which is applied in Step 3 of our main algorithm below.

Algorithm 1 *Fast Adaptive Mean Shift* [14]
Input: *A positive integer k, n data point $\mathbf{x_i}$, where i = 1, 2, ..., n, one of the data points, $\mathbf{y_1}$, and a total number N of iterations.*
Output: *An approximate local maximum (*mode*) of the density.*

1: We use a subset of the input data for learning. {For selecting optimal values of K and L, see [14].}
2: $h_i = \|\mathbf{x_i} - \mathbf{x_{i,k}}\|_1$, where $\mathbf{x_{i,k}}$ is the k-nearest neighbor of $\mathbf{x_i}$
3: $j = 1$
4: **while** $j < N$ **do**
5: Let K, L, n data points $\mathbf{x_i}$ and $\mathbf{y_1}$ be the input for Procedure 3; compute an approximate neighbor of $\mathbf{y_1}$ with radius h_i, denoted by $U_{\mathbf{y_1}}$. Let

$$\mathbf{y_2} = \frac{\sum_{\mathbf{x_i} \in U_{\mathbf{y_1}}} \frac{1}{h_i^{d+2}} \mathbf{x_i} g(\| \frac{\mathbf{y_1} - \mathbf{X_i}}{h_i} \|^2)}{\sum_{\mathbf{x_i} \in U_{\mathbf{y_1}}} \frac{1}{h_i^{d+2}} g(\| \frac{\mathbf{y_1} - \mathbf{X_i}}{h_i} \|^2)} \tag{16}$$

6: $\mathbf{y_1} = \mathbf{y_2}$
7: $j = j + 1$
8: **end while**
9: Output $\mathbf{y_2}$

The main difference between Algorithm 1 and Procedure 1 is defined by Equations 16 and Equations 15, having the impact that Algorithm 1 is faster (however, also "more approximate" only).

Algorithm 2 *Our Main Algorithm*
Input: *Three positive integers k, N (number of iterations) and T (threshold of the number of merged points to apply one of the traditional clustering algorithms,*

[4] See C++ source code at `//www.caip.rutgers.edu/riul`.

such as kmeans or clusterdata implemented in MATLAB), n old clusters C_i*, where* $i = 1, 2, \ldots, n$.
Output: *m new clusters* G_i*, where* $i = 1, 2, \ldots, m$.

1: $C = \cup_{i=1}^{n} C_i$ and $S = \emptyset$
2: **for** For each $\mathbf{x} \in C$ **do**
3: Let k, C, $\mathbf{x}$ and N be the input for Algorithm 1; compute an approximate local maximum of the density, denoted by $\mathbf{x}'$.
4: $S = S \cup \{\mathbf{x}'\}$
5: **end for**
6: Sort S according to lexicographic order.
7: Merge duplicated points in S into a single point. Denote the resulting set by S'. {Note that each point $\mathbf{x}' \in S'$ can be associated with a set, called *associated* set of $\mathbf{x}'$, denoted by $S'_{\mathbf{x}'}$, to store the original points in C such that they are mapped, using Equation (16), to the same point $\mathbf{x}'$.}
8: **if** $|S'| > T$ **then**
9: $C = S'$ and goto Step 2
10: **end if**
11: Sort S' according to the cardinalities of associated sets of points in S'.
12: Let the last m points in S' be the initial centers, apply *kmeans* to cluster S'; resulting (new) clusters are denoted by G'_i, where $i = 1, 2, \ldots, m$.
13: **for** For each $i \in \{1, 2, \ldots, m\}$ **do**
14: Output

$$G_i = (\cup_{\mathbf{x}' \in G'_i} S'_{\mathbf{x}'}) \cup \{\mathbf{x}'\}$$

15: **end for**

In Step 12 of Algorithm 2, we may replace *kmeans* by *clusterdata*. Step 11 becomes unnecessary in this case.

4 Experimental Results

Table 1 lists approximate recovery rates and pseudo recovery rates (as defined in [26]) of Algorithm 2 when applied to the previously specified sample of synthetic data (analyzed in [27] with respect to the best possible upper bound for the recovery rate). In this table, k is one of the input parameters of Algorithm 2; K and L are optimal parameters obtained in Step 1 of Algorithm 1 which is applied in Step 3 of Algorithm 2. For the used data example we used $T = 800$ in Algorithm 2. All the experiments have at most two iterations (i.e., $N \leq 2$). k_i, c_i and p_i are short for the approximate recovery rates of *kmeans*, *clusterdata*, and the pseudo recovery rates at iteration i, where $i = 1, 2$.

Table 1 shows that Algorithm 2 produces the best recovery rate if $k = 30$.

Table 2 shows approximate recovery rates and pseudo recovery rates of Algorithm 2 for $k = 30$, $K = 30$, and $L = 5$. For the selected example of input data, we used $T = 800$ in Algorithm 2. All the experiments have at most two iterations (i.e., $N \leq 2$). k_i, c_i and p_i are short for the approximate recovery rates of *kmeans*, *clusterdata*, and pseudo recovery rates at iteration i, for $i = 1, 2$.

Table 1. Approximate recovery rates and pseudo recovery rates of Algorithm 2 for the used example of simulated astronomical data and different values of k.

k	K	L	$(k_1, p_1)(\%)$	$(k_2, p_2)(\%)$	$(c_1, p_1)(\%)$	$(c_2, p_2)(\%)$
5	28	2	(29.02, 93.94)	(23.66, 93.94)	(23.15, 69.70)	(18.19, 78.79)
6	27	2	(26.49, 93.94)		(18.47, 60.61)	
7	30	3	(26.49, 93.94)	(30.02, 90.91)	(19.47, 60.61)	(30.69, 84.85)
8	30	3	(26.49, 93.94)	(30.02, 93.94)	(19.47, 60.61)	(30.69, 84.85)
9	30	3	(28.67, 93.94)	(33.71, 90.91)	(20.71, 57.58)	(26.75, 78.79)
10	30	3	(27.43, 90.91)	(35.16, 81.82)	(22.56, 63.64)	(35.32, 84.85)
11	30	3	(27.25, 87.88)	(33.77, 84.85)	(18.51, 60.61)	(30.25, 75.76)
12	28	3	(28.55, 93.94)	(34.22, 87.88)	(21.20, 69.70)	(32.04, 81.82)
13	28	3	(28.55, 93.94)	(34.22, 87.88)	(21.20, 69.70)	(32.04, 81.82)
14	28	3	(28.55, 93.94)		(21.20, 69.70)	
15	28	3	(28.79, 90.91)	(34.87, 81.82)	(25.39, 66.67)	(33.68, 75.76)
16	30	3	(27.21, 84.85)	(29.98, 87.88)	(22.03, 63.64)	(26.00, 75.76)
17	28	3	(27.74, 93.94)		(24.05, 66.67)	
18	28	3	(30.43, 93.94)		(24.64, 69.70)	
20	27	4	(28.50, 93.94)	(31.82, 78.79)	(24.84, 75.76)	(32.53, 78.79)
25	29	3	(27.47, 84.85)	(27.85, 75.76)	(15.89, 60.61)	(28.54, 75.76)
30	30	5	(26.30, 81.82)	(37.84, 93.94)	(14.95, 54.55)	(31.28, 81.82)
40	26	5	(26.30, 81.82)		(14.95, 54.55)	
50	25	4	(29.66, 96.97)		(28.73, 69.70)	
70	28	7	(30.31, 93.94)		(26.42, 69.70)	
90	29	7	(28.41, 93.94)		(24.80, 66.67)	
110	22	6	(28.67, 81.82)		(28.13, 72.73)	
130	28	8	(24.94, 81.82)		(28.84, 6970)	
150	27	9	(26.59, 93.94)		(26.72, 69.70)	

Table 2 shows that Algorithm 2 even ensures a mean recovery rate of 35.45% (using *kmeans*) or of 35.73% (using *clusterdata*).

The best possible upper bound, estimated in Subsection 4.2 in [27] for this data set, is between 39.68% and 44.71%. This allows to conclude that the obtained mean recovery rate is a "fairly good result".

5 Conclusion

We proposed a variant of adaptive mean shift clustering. Its efficiency was illustrated by applying it for the recovery of old clusters of a fairly complex set of simulated data. The obtained recovery rate is close to the estimated best possible upper bound of recovery rates for this data set, as reported in [27].

Acknowledgement: This research is part of the project "Astrovis", research program STARE (STAR E-Science), funded by the Dutch National Science Foundation (NWO), project no. 643.200.501. The first author thanks project members for valuable comments on an earlier draft of this report.

Table 2. Approximate recovery rates and pseudo recovery rates of Algorithm 2 for the used example of simulated astronomical data and $k = 30$.

j-th run	(k_1%, p_1%)	(k_2%, p_2%)	(c_1%, p_1%)	(c_2%, p_2%)
1	(28.01, 96.97)	(38.94, 81.82)	(19.19, 63.64)	(39.59, 75.76)
2	(27.66, 90.91)	(38.25, 81.82)	(15.82, 60.61)	(39.27, 78.79)
3	(26.74, 90.91)	(33.71, 84.85)	(16.84, 57.58)	(35.82, 81.82)
4	(26.18, 87.88)	(33.49, 75.76)	(25.13, 66.67)	(33.49, 75.76)
5	(28.66, 96.97)	(30.89, 81.82)	(21.48, 75.76)	(34.62, 75.76)
6	(27.28, 78.79)	(37.87, 84.85)	(18.29, 54.55)	(33.95, 75.76)
7	(24.55, 93.94)	(33.34, 75.76)	(22.50, 63.64)	(34.43, 75.76)
8	(27.24, 87.88)	(36.03, 69.70)	(22.29, 66.67)	(37.72, 72.73)
9	(26.64, 90.91)	(35.66, 72.73)	(22.86, 63.64)	(34.74, 69.70)
10	(28.47, 90.91)	(36.29, 81.82)	(21.95, 63.64)	(33.71, 75.76)
mean	(27.14, 90.61)	(35.45, 79.09)	(20.64, 63.64)	(35.73, 75.76)

References

1. K. S. Al-Sultan and M. M. Khan. Computational experience on four algorithms for the hard clustering problem. *Pattern Recognition Letters,* 17:295–308, 1996.
2. M. R. Anderberg. *Cluster Analysis for Applications.* Academic Press, Inc., New York, 1973.
3. E. Backer. *Computer-Assisted Reasoning in Cluster Analysis,* Prentice Hall, London, 1995.
4. C. Borgelt. Prototype-based Classification and Clustering. Ph.D. Thesis, University of Magdeburg, Germany, 2006.
5. F. Camastra and A. Verri. A novel kernel method for clustering. *IEEE Trans. Pattern Anal. Mach. Intell.*, 27:801–805, 2005.
6. F. E. Clements. Use of cluster analysis with anthropological data . *American anthropologist, New Series, Part 1*, 56(2):180–199, April 1954.
7. D. Comaniciu, V. Ramesh and P. Meer. The variable bandwidth mean shift and data-driven scale selection. In Proc. *Intl. Conf. Computer Vision*, volume I, pages 438–445, July 2001.
8. D. Comaniciu and P. Meer. Mean Shift: A Robust Approach toward Feature Space Analysis. *IEEE Trans. Pattern Anal. Mach. Intell.*, 24:603–619, 2002.
9. R. C. Dubes and A. K. Jain. *Clustering Methodology in Exploratory Data Analysis.* In *Advances in Computers* (M. C. Yovits, editor), pages 113–225, Academic Press, New York, 1980.
10. B. S. Duran and P. L. Odell. *Cluster Analysis: A Survey*, Springer, New York, 1974.
11. G. Efstathiou, C. S. Frenk, S. D. M. White and M. Davis. Gravitational clustering from scale-free initial conditions. *Monthly Notices RAS*, **235**:715–748, 1988.
12. B. S. Everitt. *Cluster Analysis,* Edward Arnold, London, 1993.
13. D. Fasulo. An analysis of recent work on clustering algorithms. Technical report, University of Washington, 1999.
14. B. Georgescu, I. Shimshoni and P. Meer. Mean shift Bbased clustering in high dimensions: a texture classification example. In Proc. *IEEE Int. Conf. Computer Vision*, 2003.

15. A. Gionis, P. Indyk, and R. Motwani. Similarity search in high dimensions vis hashing. In Proc. *Int. Conf. Very Large Data Bases*, pages 518–529, 1999.
16. J. A. Hartigan. *Clustering Algorithms*, John Wiley, New York, 1975.
17. A. Helmi and P. T. de Zeeuw. Mapping the substructure in the Galactic halo with the next generation of astrometric satellites. *Astron. Soc.*, **319**:657–665, 2000.
18. P. Indyk and R. Motwani. Approximate nearest neighbors: towards removing the curse of dimensionality. In Proc. *Symp. Theory Computing*, pages 604–613, 1998.
19. N. C. Jain, A. Indrayan, and L. R. Goel. Monte Carlo comparison of six hierarchical clustering methods on random data. *Pattern Recognition,* 19:95–99, 1986.
20. A. K. Jain and R. C. Dubes. *Algorithms for Clustering Data*, Prentice Hall, Englewood Cliffs, 1988.
21. A. K. Jain and P. J. Flynn. Image segmentation using clustering. In N. Ahuja and K. Bowyer, editors, *Advances in Image Understanding: A Festschrift for Azriel Rosenfeld,* pages. 65–83, IEEE Computer Society Press, 1996.
22. A.K. Jain, M.N. Murty, and P.J. Flynn. Data clustering: A review. *ACM Computing Surveys*, 31(**3**):264–323, September 1999.
23. A. Knebe, S.P.D. Gill, D. Kawata and B. K. Gibson. Mapping substructures in dark matter haloes. *Astron. Soc.*, **357**:35–39, 2005.
24. H. C. Law. Clustering, Dimensionality Reduction, and Side Information. Ph.D. Thesis, Michigan State University, the United States, 2006.
25. R. C. T. Lee. Cluster Analysis and Its Applications. In J. T. Tou, editor, *Advances in Information Systems Science,* Plenum Press, New York, 1981.
26. F. Li and R. Klette. Recovery rate of clustering algorithms. Technical Report CITR-TR-223, Computer Science Department, The University of Auckland, Auckland, New Zealand, 2008 (www.citr.auckland.ac.nz).
27. F. Li and R. Klette. About the calculation of upper bounds for cluster recovery rates. Technical Report CITR-TR-224, Computer Science Department, The University of Auckland, Auckland, New Zealand, 2008 (www.citr.auckland.ac.nz).
28. S. K. Mishra and V. V. Raghavan. An empirical study of the performance of heuristic methods for clustering In *Pattern Recognition in Practice* (E. S. Gelsema and L. N. Kanal, editors), pages 425–436, 1994.
29. B.D. Ripley. *Statistical Inference for Spatial Processes*. Cambridge University Press, 1988.
30. B.W. Silverman. *Density Estimation*. Chapman & Hall, London, 1986.
31. P. H. A. Sneath and R. R. Sokal. *Numerical Taxonomy*, Freeman, San Francisco, 1973.
32. H. Spath. *Cluster Analysis Algorithms for Data Reduction and Classification*, Ellis Horwood Publishers, England, 1980.
33. K.L. Wu and M.S. Yang. A cluster validity index for fuzzy clustering. *Pattern Recognition Lett.*, 26:1275–1291, 2005.
34. K.L. Wu and M.S. Yang. Mean shift-based clustering. *Pattern Recognition*, 40:3035–3052, 2007.
35. M.S. Yang and K.L. Wu. A similarity-based robust clustering method. *IEEE Trans. Pattern Anal. Mach. Intell.*, 26:434–448, 2004.
36. M.S. Yang and K.L. Wu. A modified mountain clustering algorithm. *Pattern Anal. Appl.*, 8:125–138, 2005.