# Recovery Rate of Clustering Algorithms

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 article provides a simple and general way for defining the recovery rate of clustering algorithms using a given family of old clusters for evaluating the performance of the algorithm when calculating a family of new clusters.
Under the assumption of dealing with simulated data (i.e., known old clusters), the recovery rate is calculated using one proposed exact (but slow) algorithm, or one proposed approximate algorithm (with feasible run time).

## 1 Introduction

Clustering has many applications in image or video analysis, such as segmentation (e.g., see [14]), learning (e.g, see [23]), bags-of-features representations of images (e.g., see [5]), or video retrieval (e.g., see [20]) - just to cite (by random selection) four examples within a large diversity of clustering applications in this area.

In general, there are hundreds of clustering algorithms proposed in the literature, often applicable in a wide diversity of areas such as computer networks, data mining, image or video analysis, and so forth. For estimating the total number of clustering algorithms, see page 5 in [13], page 13 in [17] or page 130 in [22]. This all illustrates that clustering problems are very important, and often also difficult to solve. Clustering describes unsupervised learning in its most general form.

Clustering not only interests computing scientists but also, for example, astronomers [9, 11, 15]: 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? White dwarfs and red giants were one of the great discoveries in astronomy [12].

This paper focuses on a general evaluation of clustering algorithms. Previous methods, such as [1, 3, 4, 6, 8, 18, 25], all restrict on evaluating a very small subset of clustering algorithms, while the approaches of [2] (Section 7.2.2, pages 221–222) and [21] are more complicated. Since there is a huge number of clustering algorithms, it is very important to design a simple evaluation method in order to choose a suitable clustering algorithm for a given data set. This paper proposes a sound, general and simple method to evaluate the performance of an arbitrary clustering algorithm.

For example, if a given image segmentation task may be described by simulated data, then the provided method can be used for comparing various clustering techniques designed for solving this image segmentation task.

The rest of this paper is organized as follows: Section 2 gives our definition of recovery rate. Section 3 describes algorithms for computing recovery rate of a clustering algorithm with respect to simulated input data. Section 4 gives some examples to illustrate the computation of recovery rate. Section 5 shows the experimental results. Section 6 concludes the paper.

## 2 Definitions

A definition of recovery rate is needed for having a sound measure when comparing clustering techniques.

Let $d$, $n$, and $n_i$ be positive integers, for $i = 1, 2, \ldots, n$. Let $x_{i_j} \in \mathbb{R}^d$, where $i = 1, 2, \ldots, n$ (the index of a cluster) and $j = 1, 2, \ldots, n_i$ (the subindex of a point in cluster $i$ which contains $n_i$ points). $x_{i_j}$ is called a *dD data point*.[3]

Let $C_i = \{x_{i_j} : j = 1, 2, \ldots, n_i\}$, for $i = 1, 2, \ldots, n$. $C_i$ is called a *cluster* of $d$D data points. Each cluster $C_i$ is uniquely identified by its *cluster ID*. Here we simply take index $i$ to be the cluster ID of cluster $C_i$. Clusters are always assumed to be pairwise disjoint. (We do not address fuzzy clustering in this paper.)

**Definition 1.** *A* clustering algorithm*, denoted by $\mathcal{A}$, is an algorithm which maps a finite set of points of $\mathbb{R}^d$ into a family of clusters.*

We consider the case where the union

$$C = \cup_{i=1}^{n} C_i, \text{ with } N = \operatorname{card} C,$$

of the given family of *old clusters* defines the input of a clustering algorithm, without having any further information about points in this set, such as their cluster ID. The output is a partition of this union $C$ into a finite number of *new clusters* $G_k$, where $k = 1, 2, \ldots, m$:

$$\cup_{k=1}^{m} G_k = C$$

A new cluster $G_k$ may contain data points from different old clusters, and we partition $G_k$ into subsets based on the cluster ID of the old clusters:

$$G_k = \cup_{t_k=1}^{s_k} G_{k_{t_k}}$$

where $G_{k_{t_k}}$ is a subset of an old cluster, for $t_k = 1, 2, \ldots, s_k$.

[3] We prefer to write $x_{i_j}$ rather than $x_{ij}$, for indicating that $j$ is a subindex in the cluster identified by index $i$.

Obviously, indices $i$ and $k$ are in some permutation (old cluster $C_i$ is not necessarily 'more related' to new cluster $G_i$ than to any other new cluster $G_k$), and we may even have that $n \neq m$.

For defining the recovery rate, we assume that each old cluster can only be represented by one new cluster, defining a mapping $i \to k$, and for this $k$ we then have the defined subset $G_{k_{i_k}}$ which should have maximum cardinality compared to all the other $G_{k_{t_k}}$ of new cluster $G_k$. However, for defining the recovery rate we have to detect the maximum of all possible global mappings $i \to k$; just a value for one particular $i$ will not do.

**Definition 2.** *Assume that $G_{1_{t'_1}}$, $G_{2_{t'_2}}$, ..., $G_{m_{t'_m}}$ satisfy*

*(i) For $i$, $j \in \{1_{t'_1}, 2_{t'_2}, \ldots, m_{t'_m}\}$, there exist two old clusters $C_i$ and $C_j$ such that $G_{i_{t'_i}} \subseteq C_i$ and $G_{j_{t'_j}} \subseteq C_j$; and*

*(ii)* $\sum_{k=1}^{m} \frac{cardG_{k_{t'_k}}}{cardC_k} = \max\{\sum_{k=1}^{m} \frac{cardG_{k_{t_k}}}{cardC_k} : t_k = 1, 2, \ldots, s_k\}$

*The value*

$$\frac{\sum_{k=1}^{m} \frac{cardG_{k_{t'_k}}}{cardC_k}}{m} \times 100\%$$

*is called the* recovery rate *of the clustering algorithm $\mathcal{A}$ with respect to the input $\cup_{i=1}^{n} C_i$.*

Definition 2 assumes that we have $m \leq n$; the number of new clusters is upper bounded by the the number of old ones. We do not consider this as a crucial restriction of generality.

It is obvious that if all $cardC_i$ are identical, where $i = 1, 2, \ldots, n$, then item (ii) in Definition 2 can be simplified as follows:

$$\sum_{k=1}^{m} cardG_{k_{t'_k}} = \max\{\sum_{k=1}^{m} cardG_{k_{t_k}} : t_k = 1, 2, \ldots, s_k\}$$

And the recovery rate of the clustering algorithm with respect to the input is the value below:

$$\frac{\sum_{k=1}^{m} cardG_{k_{t'_k}}}{\sum_{i=1}^{n} cardC_i} \times 100\%$$

Obviously, there can be further options of defining a recovery rate, for example by enforcing $m = n$ and always comparing $G_i$ with $C_i$, but we consider the above definition as the least restrictive. We only mention one more option here for (possibly) defining a recovery rate:

**Definition 3.** *Assume that $G_{1_{t_1}}$, $G_{2_{t_2}}$, ..., $G_{m_{t_m}}$ satisfy*

*(i) For $i$, $j \in \{1_{t_1}, 2_{t_2}, \ldots, m_{t_m}\}$, there exist two old clusters $C_i$ and $C_j$ such that $G_{i_{t_i}} \subseteq C_i$ and $G_{j_{t_j}} \subseteq C_j$;*
*(ii) $G_{i_{t_i}} \neq \emptyset$, where $i \in \{1, 2, \ldots, m\}$; and*
*(iii) $m$ is maximal.*

*The value*

$$\frac{m}{n} \times 100\%$$

*is called the* pseudo recovery rate *of the clustering algorithm $\mathcal{A}$ with respect to the input $\cup_{i=1}^{n} C_i$.*

Our definitions are very intuitive (and certainly easy to understand). Our method does not need to introduce other functions such as an $F$-function as in [16], or entropy as in [2] or [4].

In the next section we will illustrate that the pseudo recovery rate is actually not a reasonable choice, and we will then only apply recovery rate as defined in Definition 2 afterwards.

## 3 Algorithms

This section assumes simulated data, such that both the old and new clusters are known.

We propose two different algorithms and discuss their properties afterwards, also for the purpose of comparing both with one-another. The first algorithm is straightforward, but computationally expensive:

**Algorithm 1: Exact Recovery Rate**

Input: Old clusters $C_i$, where $i = 1, 2, \ldots, n$; and new clusters $G_j$, where $j = 1, 2, \ldots, m$, obtained from a clustering algorithm $\mathcal{A}$.

Output: The recovery rate of $\mathcal{A}$ with respect to $C_i$, where $i = 1, 2, \ldots, n$.

1. Let $M$ be an $m \times n$ matrix, initially with zeros in all of its elements.
2. For each $j \in \{1, 2, \ldots, m\}$ and for each $x \in G_j$, if there exists an $i \in \{1, 2, \ldots, n\}$ such that $x \in C_i$, then update $M$ as follows: $M(j, i) = M(j, i) + 1$, where $M(j, i)$ is the $(j, i)$-th entry of $M$.
3. Find $m$ different integers (i.e., column indices) $i_k \in \{1, 2, \ldots, n\}$ such that

$$\sum_{k=1}^{m} \frac{M(k, i_k)}{cardC_{i_k}} = \max\{\sum_{j=1}^{m} \frac{M(j, i_j)}{cardC_{i_j}} : i_j \in \{1, 2, \ldots, n\}\}$$

4. Output the recovery rate as being the value

$$\frac{\sum_{k=1}^{m} \frac{M(k, i_k)}{cardC_{i_k}}}{m} \times 100\%$$

The main computations of Algorithm 1 occur in Step 3, and its time complexity (note: $m \leq n$) equals

$$O(n(n-1)\cdots(n-m+2)(n-m+1)) \geq O(m!) \geq O(2^m)$$

Obviously, this exponential time algorithm calculates the correct recovery rate.

The following is only an approximate algorithm for computing the recovery rate, but with feasible running time.

**Algorithm 2: Approximate Recovery Rate**

Input and Steps 1 and 2 are the same as in Algorithm 1.

Output: The approximate recovery rate of $\mathcal{A}$ with respect to $C_i$, where $i =$ 1, 2, …, $n$.

3.0. For each entry $M(i,j)$ of $M$, let $M(i,j) = \frac{M(i,j)}{cardC_j}$, where $i =$ 1, 2, …, $m$; $j =$ 1, 2, …, $n$.

3.1. For each $j \in \{1, 2, \ldots, m\}$, find the maximum entry of $M$, denoted by $m_j = M(i,k)$.

3.2. Update $M$ by removing the $i$-th row and $k$-th column of $M$ and go to Step 3.1.

4. Output the approximate recovery rate as the value

$$\frac{\sum_{j=1}^{m} m_j}{m} \times 100\%$$

It follows that the approximate recovery rate, obtained from Algorithm 2, is less than or equal to the exact recovery rate obtained from Algorithm 1. The time complexity of Algorithm 2 is in $\mathcal{O}(mn)$.

Both of our algorithms are very simple to implement. We only use a single matrix, while [2] uses several matrices.

## 4 Examples

The first two examples are on purpose easy to follow such that the reader may follow the proposed definitions and algorithms. Let

$$C_1 = \{(1,5,5), (5,9,3), (6,9,6), (7,6,4), (9,6,0)\}$$

and

$$C_2 = \{(7,14,13), (10,15,12), (14,16,7), (15,7,16), (16,7,12)\}$$

be two clusters of 3D data points (see Figure 1, left).

$$C_1 \cup C_2 = \{(1,5,5),(5,9,3),(6,9,6),(7,6,4),(7,14,13),(9,6,0),(10,15,12),\\ (14,16,7),(15,7,16),(16,7,12)\}$$

is the union of $C_1$ and $C_2$ (see Figure 1, middle).

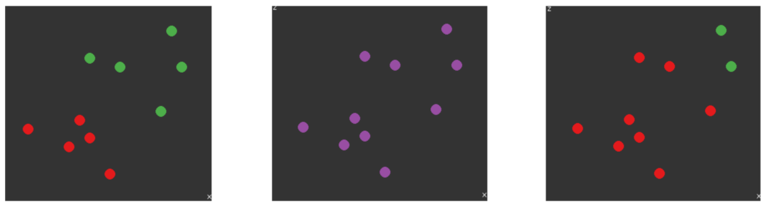


**Fig. 1.** Left: red points belong to $C_1$; green points belong to $C_2$. middle: the union of $C_1$ and $C_2$. right: red points belong to $G_1$; green points belong to $G_2$.

Let $\mathcal{A}$ be the algorithm for clustering data in MATLAB$^{TM}$, called *clusterdata*. The obtained output is $G_1 = \{\ (5,9,3),(7,6,4),(6,9,6),(9,6,0),(1,5,5),(10,15,12),(14,16,7),(7,14,13)\ \}$, and $G_2 = \{(15,7,16),(16,7,12)\}$ (see Figure 1, right).

Let $G_1 = G_{1_1} \cup G_{1_2}$, where $G_{1_1} = \{(5,9,3),(7,6,4),(6,9,6),(9,6,0),(1,5,5)\}$, $G_{1_2} = \{(10,15,12),(14,16,7),(7,14,13),\}$; $G_1 = G_{2_1}$, where $G_{2_1} = \{(15,7,16),(16,7,12)\}$.

*Example 1.* By Algorithm 1, we have that

$$M = \begin{pmatrix} 5 & 3 \\ 0 & 2 \end{pmatrix}$$

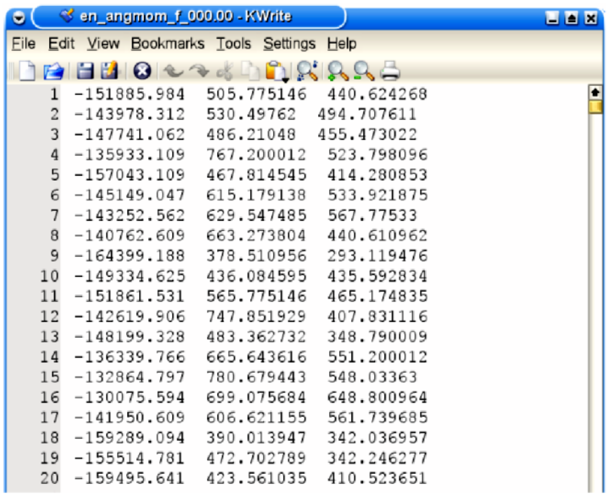
In Step 3 of Algorithm 1, there are only two cases to select different column indices: $i_1 = 1$ and $i_2 = 2$ or $i_1 = 2$ and $i_2 = 1$. Thus, the recovery rate of $\mathcal{A}$ with respect to $C_1 \cup C_2$ is equal to

$$(M(1,1) + M(2,2))/|G_1 \cup G_2| \times 100\% = (5+2)/10 \times 100\% = 70\%$$

*Example 2.* By Algorithm 2, we have the same matrix $M$ as in Algorithm 1. We obtain that $m_1 = 5$ and $m_2 = 2$ in Step 3 of Algorithm 2. Thus, the approximate recovery rate of $\mathcal{A}$ with respect to $C_1 \cup C_2$ is equal to

$$(m_1 + m_2)/|G_1 \cup G_2| \times 100\% = (5+2)/10 \times 100\% = 70\%$$

So far about these simple two examples, where both algorithms actually produce the same result (i.e., recovery rate). The next two examples show that Algorithms 1 and 2 could produce different results.


en_angmom_f_000.00 - KWrite
File Edit View Bookmarks Tools Settings Help

```
-151885.984  505.775146  440.624268
-143978.312  530.49762  494.707611
-147741.062  486.21048  455.473022
-135933.109  767.200012  523.798096
-157043.109  467.814545  414.280853
-145149.047  615.179138  533.921875
-143252.562  629.547485  567.77533
-140762.609  663.273804  440.610962
-164399.188  378.510956  293.119476
-149334.625  436.084595  435.592834
-151861.531  565.775146  465.174835
-142619.906  747.851929  407.831116
-148199.328  483.362732  348.790009
-136339.766  665.643616  551.200012
-132864.797  780.679443  548.03363
-130075.594  699.075684  648.800964
-141950.609  606.621155  561.739685
-159289.094  390.013947  342.036957
-155514.781  472.702789  342.246277
-159495.641  423.561035  410.523651
```

**Fig. 2.** Some data points in cluster 0 (which is stored in the text file en_angmom_f_000.00 [7]).

We combine an adaptive mean shift based clustering algorithm (see [10]) with traditional clustering algorithm *kmeans* (another clustering algorithm in MATLAB) to obtain a variant of mean shift based clustering algorithm, denoted by $\mathcal{K}$.

We illustrate problems of clustering (for easier illustration) in the following two examples for a simulation of astronomical data (publicly available on [7]) rather than for some examples of image or video data. Clusters in those astronomical data (further illustrated in Section 5) are characterized by being highly overlapping. Obviously, the recovery of highly overlapping data is difficult, if not even (nearly) impossible. Even currently published cluster algorithms (see, for example, [24]) work neither efficiently nor correctly.

There are 10,000 3D data points in each cluster (which is stored in a text file named “en_angmom_f_000.i”, where $i$ = 00, 01, 02, 04, and 05). For example, Figure 2 shows the first 20 data points in cluster 0 (i.e., in the file en_angmom_f_000.00).

The union of these five old clusters is shown in Figure 3.

*Example 3.* By Algorithm 1, we have that

$$M = \begin{pmatrix} 1612 & 0 & 24 & 2009 & 0 \\ 0 & 0 & 21 & 0 & 4540 \\ 0 & 0 & 4153 & 2 & 5460 \\ 0 & 10000 & 5796 & 7989 & 0 \\ 8388 & 0 & 6 & 0 & 0 \end{pmatrix}$$

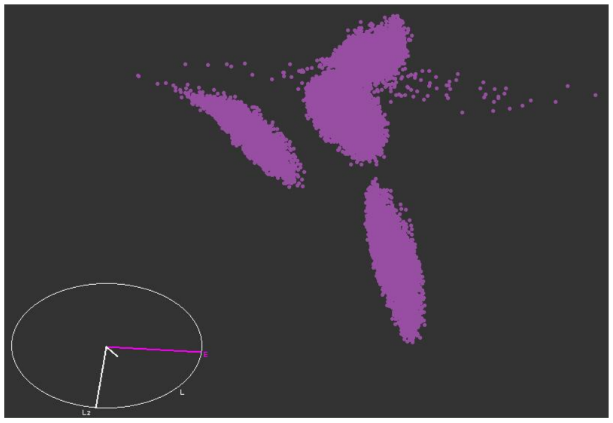


**Fig. 3.** An example of an overlapping data set: This shows a 2D projection of a union of 5 clusters, where each cluster contains 10,000 3D data points [11].

Thus, the recovery rate of $\mathcal{K}$ with respect to the input data shown in Figure 3 is equal to

$$(M(1,4) + M(2,5) + M(3,3) + M(4,2) + M(5,1))/5 * 10000 \times 100\%$$

$$= (2009 + 4540 + 4153 + 10000 + 8388)/50000 \times 100\% = 58.18\%$$

*Example 4.* By Algorithm 2, we have the same matrix $M$ as in the previous example. Thus, the approximate recovery rate of $\mathcal{K}$ with respect to the input data shown in Figure 3 is equal to

$$(M(4,2) + M(5,1) + M(3,5) + M(1,4) + M(2,3))/5 * 10000 \times 100\%$$

$$= (10000 + 8388 + 5460 + 2009 + 21)/50000 \times 100\% = 51.76\%$$

Examples 1 to 4 illustrate that Algorithms 1 and 2 may lead to different results with respect to the recovery rate (as in Definition 2). Another difference in evaluations, using either Algorithm 1 or 2, is that Algorithm 1 produces the exact recovery rate but it has time complexity $\mathcal{O}(2^m)$ while Algorithm 2 produces an approximate recovery rate but it has the time complexity $\mathcal{O}(mn)$.

The definition of recovery rate allows us to measure the ability of a clustering algorithm with respect to given input data. It also allows us to compare the performance of two different clustering algorithms. For example, it is simple to compute the recovery rate of *kmeans* with respect to $C_1 \cup C_2$ in Examples 1 and 2, and it equals 100%. So we may say that *kmeans* is better than *clusterdata* for this input.

Finally, we illustrate the failure of pseudo recovery rate to provide a proper value.

*Example 5.* Since $G_{1_1} \neq \emptyset$ and $G_{2_1} \neq \emptyset$, by Definition 3, the pseudo recovery rate of $\mathcal{A}$ with respect to $C_1 \cup C_2$ is equal to

$$\frac{2}{2} \times 100\% = 100\%$$

Example 5 illustrates the flaw when using the defined pseudo recovery rate to evaluate the performance of clustering algorithms: Even though the pseudo recovery rate is 100%, it does not mean that all the old clusters have been recovered completely. In particular, the use of pseudo recovery rate will exaggerate claims when the cardinalities of old clusters are very large, such as (typically) in the case of clustering a union of a number of galaxies in astronomy.

Altogether, this should be sufficient to illustrate our point of view that clustering results need to be evaluated with respect to any possible mapping of all the generated $m$ new clusters into the set of all available $n$ old clusters.

## 5 Experimental Results

We combine an adaptive mean shift based clustering algorithm (see [10]) with traditional clustering algorithm *kmeans* (or *clusterdata*) to obtain a variant of mean shift based clustering algorithm, denoted by $\mathcal{K}$ (or $\mathcal{C}$). We continue with the astronomical data as used in Examples 3 and 4.

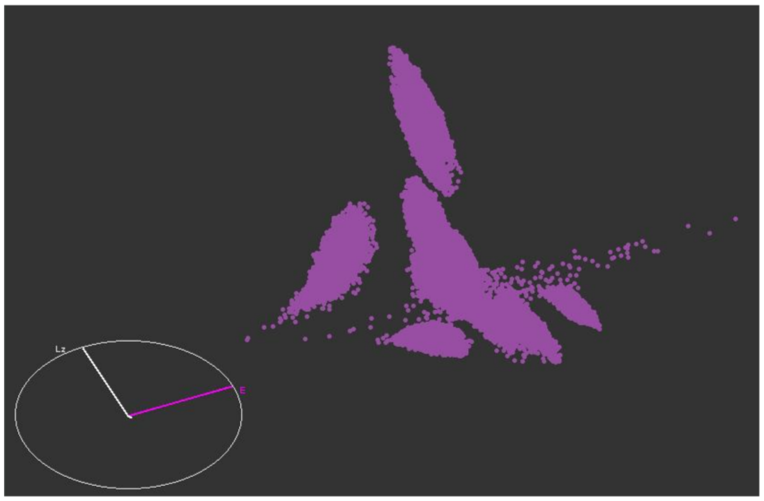


**Fig. 4.** An example of a very heavily overlapping data set: This shows a 2D projection of a union of 10 clusters, where each cluster contains 10,000 3D data points [11].

### 5.1 The Input Data Set

There are 10,000 3D data points in each cluster of the data set on [7] (which is stored in a text file named "en_angmom_f_000.i", where $i = 00, 01, 02, \ldots, 09, 10, \ldots, 32$). For example, Figure 2 shows the first 20 data points in cluster 0 (i.e., in the file en_angmom_f_000.00). The union of the first 10 clusters is shown in Figure 4.

Input data used in experiment below refer to this data set, but after the following normalization (just for scale reduction): For each point $p = (x, y, z)$ in the data set, replace $p$ by $(x/20, y/11, z/11)$.

### 5.2 Some Results

Tables 1 and 2 show recovery rates, approximate recovery rates, and pseudo recovery rates of Algorithms $\mathcal{K}$ and $\mathcal{C}$. $n$ is the number of old clusters of the input data in Section 5.1 (i.e., the first $n$ old clusters of the 33 old clusters). We use either one (Table 1) or two (Table 2) iterations. $k_1$ ($c_1$) is the recovery rate of Algorithm $\mathcal{K}$ ($\mathcal{C}$), which is obtained by Algorithm 2. $k_2$ ($c_2$) is the approximate recovery rate of Algorithm $\mathcal{K}$ ($\mathcal{C}$), which is obtained by Algorithm 1. $p$ is short

**Table 1.** This table shows the results of Iteration 1.

| $n$ | ($k_1$%, $t_1$ sec, $k_2$%, $t_2$ sec, $p$%) | ($c_1$%, $t_1$ sec, $c_2$%, $t_2$ sec, $p$%) |
|---|---|---|
| 5 | (59.4, 6.2e-4, 59.4, 4.3e-3, 100) | (39.6, 8.8e-4, 39.6, 4.3e-3, 80) |
| 6 | (51.6, 7.2e-4, 56.0, 4.6e-2, 100) | (40.2, 7.0e-4, 40.2, 2.7e-2, 66.7) |
| 7 | (58.7, 8.4e-4, 58.7, 0.3, 100) | (28.9, 9.1e-4, 28.9, 0.3, 57.1) |
| 8 | (44.5, 0.01, 44.5, 2.3, 87.5) | (15.7, 9.9e-4, 15.7, 2.4, 50) |
| 9 | (47.6, 1e-3, 47.6, 23.2, 88.9) | (12.5, 0.3, 12.5, 23.3, 44.4) |
| 10 | (44.4, 0.4, 44.9, 285.1, 90) | (24.7, 1.2e-3, 24.7, 256.5, 40) |

**Table 2.** This table shows the results of Iteration 2.

| $n$ | ($k_1$%, $t_1$ sec, $k_2$%, $t_2$ sec, $p$%) | ($c_1$%, $t_1$ sec, $c_2$%, $t_2$ sec, $p$%) |
|---|---|---|
| 5 | (51.8, 6.0e-4, 58.2, 4.3e-3, 100) | (36.8, 5.9e-4, 36.8, 5.0e-3, 60) |
| 6 | (66.2, 6.7e-4, 66.2, 2.9e-2, 100) | (46.6, 6.9e-4, 46.6, 3.4e-2, 83.3) |
| 7 | (57.4, 7.6e-4, 57.4, 0.3, 71.4) | (39.2 , 7.6e-4, 39.2, 0.3, 85.7) |
| 8 | (49.9, 8.6e-4, 49.9, 2.3, 87.5) | (42.8, 1.1e-3, 42.8, 2.4, 75) |
| 9 | (46.7, 3.3e-3, 46.7, 23.6, 77.8) | (26.3, 0.01, 26.3, 23.1, 66.7) |
| 10 | (53.4, 9.1e-3, 53.4, 303.1, 90) | (51.0, 1.7e-3, 51.0, 272.8, 80) |

for pseudo recovery rate. $t_i$ is the running time for obtaining $k_i$ $(c_i)$, where $i = 1, 2$.

Tables 1 and 2 show that Algorithm 2 is a good approximation to Algorithm 1 when $n \leq 10$. They also illustrate that the running time of Algorithm 1 is indeed significantly longer than that of Algorithm 2 when $n \geq 10$.

## 6 Conclusions

In conclusion, in this paper we defined a recovery rate of unconstrained clustering, and provided a time-efficient approximate algorithm for estimating this recovery rate. We are now ready to compare the performance of any two clustering algorithms by comparing their recovery rates for a simulated input (for example, assuming that a clustering task in image or video analysis allows to have quite realistic simulated input data). In particular we may analyze next lower bounds for recovery rates of clustering; see [19].

## 7 Acknowledgement

The first author thanks Prof. A. Helmi for providing the url of the simulated astronomical data set (on the public web site [7]), and acknowledges that his 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.

## References

1. J. Allan, A. Feng, and A. Bolivar. Flexible Intrinsic Evaluation of Hierarchical Clustering for TDT. In Proc. *CIKM'03*, November 3–8, New Orleans, Louisiana, USA.
2. C. Borgelt. Prototype-based Classification and Clustering. Ph.D. Thesis, University of Magdeburg, Germany, 2006.
3. S. Brohee and J. van Helden. Evaluation of clustering algorithms for protein-protein interaction networks. *BMC Bioinformatics*, **7**:488, 2006.
4. D. Crabtree, X. Gao, P. Andreae. Universal Evaluation Method for Web Clustering Results. Technical Report CS-IR-05-3, Department of Computer Science, Victoria University of Wellington, New Zealand, 2005.
5. G. Csurka, C. Dance, L. Fan, J. Willamowski, and C. Bray. Visual categorization with bags of keypoints. In Proc. *ECCV Workshop Statistical Learning Computer Vision*, pages 59–74, 2004
6. S. Datta and S. Datta. Evaluation of clustering algorithms for gene expression data *BMC Bioinformatics*, **7**:(Suppl 4):S17, 2006.
7. Simulated astronomical data.
`//www.astro.rug.nl/~ahelmi/simulations\_gaia.tar.gz`
8. S. Datta and S. Datta. Methods for evaluating clustering algorithms for gene expression data using a reference set of functional classes. *BMC Bioinformatics*, **7**:397, 2006.

9. 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, Dec. 1988.
10. B. Georgescu, I. Shimshoni and P. Meer. Mean Shift Based Clustering in High Dimensions: A Texture Classification Example. In Proc. *9th IEEE International Conference on Computer Vision (ICCV)*, 2003.
11. 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.
12. Hertzsprung-Russell. `en.wikipedia.org/wiki/Hertzsprung-Russell_diagram`.
13. A.K. Jain, M.N. Murty, and P.J. Flynn. Data clustering: A review. *ACM Computing Surveys*, 31(**3**):264–323, September 1999.
14. N. Kehtarnavaz, J. Monaco, J. Nimtschek, and A. Weeks. Color image segmentation using multi-scale clustering. In Proc. IEEE Southwest Symp. Image Analysis Interpretation, pages 142–147, 1998.
15. A. Knebe, S.P.D. Gill, D. Kawata and B. K. Gibson. Mapping substructures in dark matter haloes. *Astron. Soc.*, **357**:35–39, 2005.
16. B. Larsen and C. Aone. Fast and Effective Text Mining Using Linear Time Document Clustering. In Proc. *5th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, San Diego, California: ACM Press, 16–22, 1999.
17. H. C. Law. Clustering, Dimensionality Reduction, and Side Information. Ph.D. Thesis, Michigan State University, the United States, 2006.
18. A. V. Leouski and W. B. Croft. An Evaluation of Techniques for Clustering Search Results. Technical Report IR-76, Department of Computer Science, University of Massachusetts, Amherst, 1996.
19. 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).
20. N.-X. Lian, Y.P. Tan and K.L. Chan. Efficient video retrieval using shot clustering and alignment. In Proc. *ICICS-PCM*, pages 1801–1805, 2003.
21. W. M. Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66:846–850, December, 1971.
22. B.W. Silverman. *Density Estimation.* Chapman & Hall, London, 1986.
23. Z. Wang, S.C. Chen, and T. Sun. MultiK-MHKS: a novel multiple kernel learning algorithm, *IEEE PAMI*, **30**:348–353, 2008.
24. K.L. Wu and M.S. Yang. Mean shift-based clustering. *Pattern Recognition*, 40:3035–3052, November, 2007.
25. Y. Zhao and G. Karypis. Evaluation of hierarchical clustering algorithms for document datasets. In Proc. *CIKM'02*, November 4–9, McLean, Virginia, USA.