# About the Calculation of Upper Bounds for Cluster Recovery Rates

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.** Obtaining a reasonable upper bound of the recovery rate of an arbitrary clustering algorithm is of importance when exploring clustering algorithms with respect to possible recovery rates. This paper estimates the best possible recovery rate of an arbitrary clustering algorithm with respect to any given input data set, based on two hypotheses. For an example of a reasonably complex data set, obtained results are verified and adjusted using a data visualization system.

**Key words:** clustering, recovery rate

## 1 Introduction

This paper uses a definition of recovery rate of clustering algorithms as introduced in [9]. Although there is a vast collection of clustering algorithms proposed in the literature [6, 8, 10], it seems that there does not exist any article so far which discusses upper bounds for recovery rate estimations.

Recovery rate estimation is very important, especially when clustering strongly overlapping data sets with relatively low recovery rates [1]. For example, for a very strongly overlapping data set, we should ask ourself a very important question: does there exist a clustering algorithm at all such that it may have a recovery rate of, say, $\geq 50\%$ with respect to the given input? Obtaining a reasonable (lowest) upper bound of the recovery rate of an arbitrary clustering algorithm would help algorithm designers to stop wasting time to explore clustering algorithms with respect of an actually unreachable recovery rate.

We propose in this paper two hypotheses which the reader will hopefully accept as being fairly general. Based on those two hypotheses, we then apply a digital geometry [7] technique to estimate the upper bound of the recovery rate of any (arbitrary) clustering algorithm. The rest of this paper is organized as follows: Section 2 gives some mathematical definitions of important notions such as "overlapping", used in the rest of the paper. Section 3 describes the algorithm to approximately estimate the best recovery rate. Section 4 presents experimental results for a non-trivial input example from [5]. Section 5 concludes the paper.

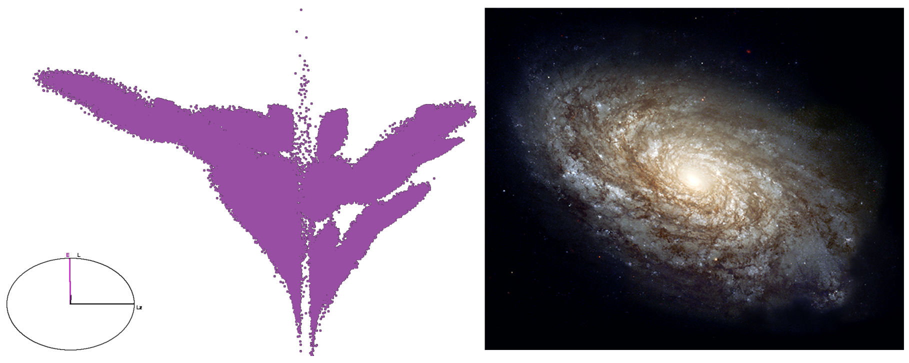

**Fig. 1.** Left: example of a synthetic data set of (partially) very heavily overlapping clusters: The figure shows a 2D projection of a union of 33 clusters, where each cluster contains 10,000 3D data points [5]. Right: a typical spiral galaxy (NGC 4414); source: ESA and NASA.

Figure 1 illustrates an example of simulated data as used in [5], and an image of a spiral galaxy, being the application background for this synthetic data set. These simulated astronomical data are available on the public web site [2]. This is a reasonably complex data set, and we will use it for illustration in this paper. For example, see Figure 2 for an illustration of two 'adjacent' clusters, contributing to the shown union of all 33 clusters. (We will show that a cluster such as $C_{16}$ may have a high recovery rate which ranks[3] 30 out of all 33 clusters). However, the proposed approach is independent of this example of 33 clusters, used here for demonstrating purposes only.

## 2 Definitions

This section defines basic notions used to describe our algorithm in the next section. In this paper, any mentioned distance refers to Euclidean distance, and a set is always finite.

**Definition 1.** *Let $S$ be a set of dD data points and $p$ a point in $S$. The value*

$$\min\{d(p,q) : q \in S\backslash\{p\}\}^4$$

*is called the* outlier distance *(with respect to $p$ and $S$), denoted by $d(p, S\backslash\{p\})$; the value* $\min\{d(p, S\backslash\{p\}) : p \in S\}$ *(*$\max\{d(p, S\backslash\{p\}) : p \in S\}$ *) is called the* minimum (maximum) *outlier distance of $S$, denoted by $mod(S)$ ($Mod(S)$); the value*

$$\frac{\sum_{p\in S} d(p, S\backslash\{p\})}{|S|}$$

[3] The ranking is with respect to increasing recovery rate.
[4] $d(p,q)$ is the Euclidean distance between $p$ and $q$.

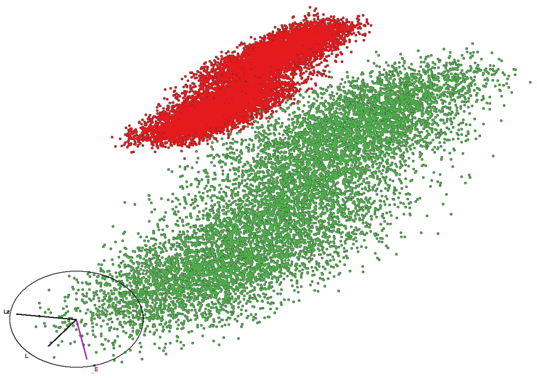
**Fig. 2.** Cluster $C_{16}$ (red) of this data set with a unique 'adjacent' cluster (green). We will see that for this example, the upper bound of the recoverable quantity of $C_{16}$ equals 8,081, and this is rank 30 of all 33 clusters.

*is called the* averaged *outlier distance of* $S$, *denoted by* $aod(S)$.

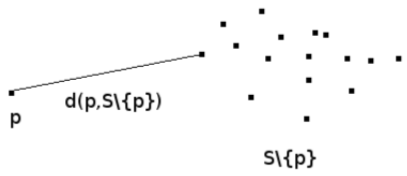

**Fig. 3.** Illustration of outlier distance.

It follows that $mod(S)$, $Mod(S)$ and $aod(S)$ are positive real numbers (not necessarily integers!). For example, let $S$ be the set of grid vertices of a finite unit grid $G$, and $p$ an arbitrary grid vertex of $G$; then, the outlier distance (with respect to $p$ and $S$) is equal to the grid constant 1 of $G$. Obviously, in this case we have that $mod(S) = Mod(S) = aod(S) = 1$.

**Definition 2.** *Let* $S_1$ *and* $S_2$ *be two sets of dD data points. Let* $d(S_1, S_2) = \min\{d(p, q) : p \in S_1 \wedge q \in S_2\}$. *If* $aod(S_1) < d(S_1, S_2)$, *then we say that* $S_1$ *does* not interfere *with* $S_2$; *otherwise* $S_1$ *does* interfere *with* $S_2$

In Figure 4, let $S_1$ and $S_2$ be the vertices of the small and big squares, respectively; $aod(S_i) = d_i$, where $i = 1, 2$; and $d(S_1, S_2) = d_3$.

Figure 4 (left) shows a case with $d_2 > d_1 > d_3$. By Definition 2, $S_1$ interferes with $S_2$, and $S_2$ interferes with $S_1$ as well.

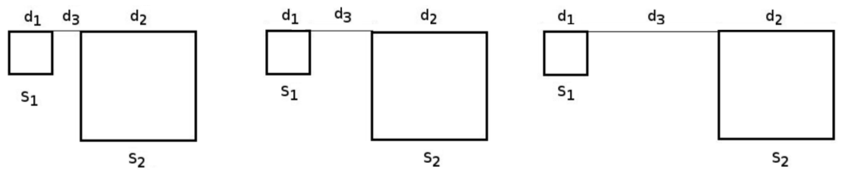

**Fig. 4.** Illustration of "does interfere with" and "does not interfere with".

Figure 4 (middle) shows a case of $d_1 < d_3$. By Definition 2, $S_1$ does not interfere with $S_2$. Also assuming $d_2 > d_3$, by Definition 2 it follows that $S_2$ interferes with $S_1$. Therefore, interfering and non-interfering may occur in a non-symmetrical way.

Figure 2 shows that for the given example of simulated astronomical data, interfering and non-interfering may occur in a non-symmetrical way. This is because distribution and density of the two clusters may differ.

Figure 4 (right) shows a case with $d_1 < d_2 < d_3$. By Definition 2, $S_1$ does not interfere with $S_2$, and $S_2$ does not interfere with $S_1$ as well.

In this paper, "interfering" and "overlapping" is interchangeable.

**Definition 3.** *Let $C_1$ and $C_2$ be two (different) old clusters of dD data points. If $C_1$ interferes with $C_2$, then $C_2$ is* adjacent *to $C_1$.*

Since the relations of "interferes with" and "does not interfere with" are not symmetrical in general, this adjacency relation is also not symmetrical in general.

**Definition 4.** *Let $d_0 \in \mathbb{R}^1$. The subset $\{p : d(p, S \backslash p) \leq d_0 \wedge p \in S\}$ is called the subset of $S$* induced *by $d_0$, denoted by $S(d_0)$. We denote $S(d_0)$ by $S_{aod}$ if $d_0 = aod(S)$.*

It follows that $S(d_0) = \emptyset$ if $d_0 < mod(S)$, and $S(d_0) = S$ if $d_0 \geq Mod(S)$.

*Example 1.* Let $S = \{\ (16,135), (195,100), (213,72), (225,90), (239,142),$ $(250,59), (256,105), (266,85), (290,159), (293,126), (293,102), (299,80),$ $(309,82), (328,103), (333,136), (352,107), (377,104)\ \}$, and $p = (16,135)$; see Figure 5.

By above definitions, we have that $d(p, S \backslash \{p\}) = 182.3897 = d(p, (195,100))$, $mod(S) = 10.1980$, $Mod(S) = 182.3897$, $aod(S) = 34.1648$, and $S_{aod}=$ $\{\ (195,100), (213,72), (225,90), (250,59), (256,105), (266,85), (290,159),$ $(293,126), (293,102), (299,80), (309,82), (328,103), (333,136), (352,107),$ $(377,104)\ \}$. It follows that $S \backslash \{S(aod(S))\} = \{(16,135), (239,142)\}$.

Let $S_1$ and $S_2$ be two sets of dD data points. If there exists a subset $S_m$ of $S_1(aod(S_1))$ such that $S_1(aod(S_1)) \backslash S_m$ does not interfere with $S_2$, and $|S_m|$ is minimal, then $|S_1| - |S_m|$ is called the *maximized recoverable quantity* of $S_1$ with respect to $S_2$.

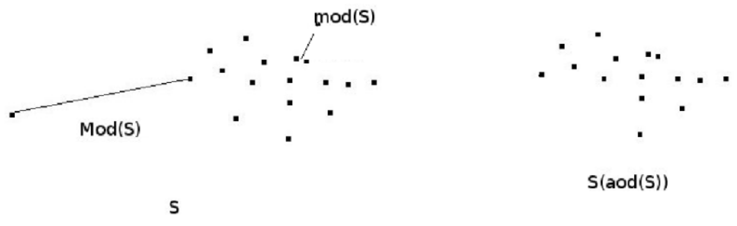

**Fig. 5.** Illustration of the values computed in Example 1. Left: $S$, $Mod(S)$ and $mod(S)$; Right: $S_{aod}$.

## 3 The Algorithm

This section describes an algorithm to estimate the upper bound of the recovery rate of an arbitrary clustering algorithm based on the idea of digital geometry. In the rest of the paper, any set to be clustered is a set of 3D data points.

### 3.1 Insert 3D Data Points into Cubes

The following Procedure 1 and its two slightly modified versions (Procedures 2 and 3) are later used in the main procedure (Procedure 4) of this subsection. The basic idea is to partition the set of given data points by insertion into a number of cubes. Assume that all cubes are of equal size and partitioning $\mathbb{R}^3$ into a uniform orthogonal grid. Each cube is identified with the coordinates of its centroid (see the grid-cube model in 3D digital geometry [7]).

A *layer* (of cubes) is a set of cubes such that one coordinate ($x$, $y$ or $z$) is constant for each cube in the set. A *strip* (of cubes) is a set of cubes such that two coordinates ($x$ and $y$, $x$ and $z$, or $y$ and $z$) are constant for all cubes in the set.

**Procedure 1** *Insert 3D Data Points into Layers*

*Input: An 1D array of 3D data points, denoted by $P$.*
*Output: A number of layers containing these input data points.*

```
1. Let $\theta$ be the grid constant, $n$ the length of $P$, and $P[i].x$ the $x$-coordinate of point $P[i]$, where $i$ = 0, 1, 2, ..., $n-1$.
2. Sort input data points in $P$ by $x$-coordinate.
3. For each $i \in \{0, 1, 2, \ldots, n-1\}$ do the following:
    3.1. Let $j$ be the integer part of the real number $\frac{P[i].x - P[0].x}{\theta}$
    3.2. Insert $P[i]$ into a set $S_j$.
4. Output all $S_j$s.
```

The following two procedures are also fairly simple (they do as specified in their headers), and not given here for that reason:

**Procedure 2** *Insert 3D Data Points (all in one Layer) into Strips*

**Procedure 3** *Insert 3D Data Points (all in one Strip) into Cubes*

Now we consider *cube objects*; a cube object is defined by its center and a grid constant. Thus, we allow variable grid constants here rather than having one universally fixed for all $\mathbb{R}^3$.

**Procedure 4** *Insert 3D Data Points into Cube Objects*[5]

*Input: An 1D array $P$ of 3D data points.*
*Output: A set $CO$ of cube objects containing the input data points.*

*1. Let $CO = \emptyset$.*
*2. Sort input data points by $x$-coordinate.*
*3. Calculate the number of layers used along the $x$-axis, denoted by $n_l$, which is the integer part of the real number*

$$\frac{P[n-1].x - P[0].x}{\theta} + 1$$

*4. Apply Procedure 1 to insert the input data points into layers $L_j$, where $j$ = 0, 1, 2, ..., $n_l$ - 1.*
*5. For each $j \in \{0, 1, 2, \ldots, n_l - 1\}$ do the following:*
*5.1. Sort the data points in $L_j$ by $y$-coordinate.*
*5.2. Calculate the number of strips used along the $y$-axis, denoted by $n_{sy}$, which is the integer part of the real number*

$$\frac{P_j[n_j - 1].y - P_j[0].y}{\theta} + 1$$

*5.3. Apply Procedure 2 to insert the data points from one layer into strips $S_j$, where $j$ = 0, 1, 2, ..., $n_{sy}$ - 1.*
*5.4. For each $k \in \{0, 1, 2, \ldots, n_{sy} - 1\}$ do the following:*
*5.4.1. Sort the data points in $S_j$ by $z$-coordinate.*
*5.4.2. Calculate the number of cubes used along the $z$-axis, denoted by $n_{cz}$, which is the integer part of the real number*

$$\frac{P_{j_k}[n_{j_k} - 1].z - P_{j_k}[0].z}{\theta} + 1$$

*5.4.3. Apply Procedure 3 to insert the data points from one strip into cubes $C_j$, where $j$ = 0, 1, 2, ..., $n_{cz}$ - 1.*
*5.4.4. For each $l \in \{0, 1, 2, \ldots, n_{cz} - 1\}$ do the following:*

[5] A cube object often contains more information than a cube does.

*5.4.4.1. Compute the center of cube $C_{j_{k_l}}$, denoted by $(x_{j_{k_l}}, y_{j_{k_l}}, z_{j_{k_l}})$, as follows:*

$$x_{j_{k_l}} = P[0].x + \theta * (j+1) - 0.5 * \theta$$

$$y_{j_{k_l}} = P_0[0].y + \theta * (k+1) - 0.5 * \theta$$

$$z_{j_{k_l}} = P_{0_0}[0].z + \theta * (l+1) - 0.5 * \theta$$

*5.4.4.2. Create a cube object $CO_{j_{k_l}}$ such that its center is $(x_{j_{k_l}}, y_{j_{k_l}}, z_{j_{k_l}})$ for grid constant is $\theta$.*
*5.4.4.3. Move all data points from $C_{j_{k_l}}$ into $CO_{j_{k_l}}$.*
*5.4.4.4. Insert $CO_{j_{k_l}}$ into the set $CO$.*
*6. Output $CO$.*

Procedure 4 is frequently called in the main procedure (Procedure 6, see below) of our clustering algorithm.

## 3.2 Estimation of the Best Recovery Rate

The following simple procedure (Procedure 5) deletes a subset of data points, contained in a second set, from the first set of data points, using the digital-cubes method. The purpose is that the resulting subset of the first set will not interfere (anymore) with the second set. This procedure is frequently called in the next procedure (Procedure 6).

**Procedure 5** *Approximate Set Difference*

*Input: Two sets of cubes, $C_1$ and $C_2$, and grid constant $\theta$.*
*Output: A subset of $C_1$, denoted by $C_1'$ such that for every $c_1 \in C_1'$ and $c_2 \in C_2$, we have that $d(c_1, c_2) > \sqrt{2}\theta$*

```
1. Let $C_1' = \emptyset$.
2. For each $c_1 \in C_1$, if $d(c_1, C_2) > \sqrt{2}\theta$, then $C_1' = C_1' \cup \{c_1\}$.
3. Output $C_1'$.
```

In the following, we make use of the following

**Hypothesis 1** *In Procedure 5, the cardinality of $C_1'$ is a decreasing function of the grid constant $\theta$.*

We have to state this as a hypothesis (and not as a lemma) because it is true only approximately. We will discuss this issue in Section 4.

The following Procedure 6 is the main procedure when dealing with two sets of 3D data points. It is based on Hypothesis 1 and uses binary search to find the best grid constant in order to compute an approximately maximized recoverable quantity. Threshold $T_1$ ($T_2$) is used to decide when to terminate the procedure (i.e., when a change in the size of the grid constant (for a subset of data points) is already relatively small).

For the example of input data, illustrated in Figure 1, the cardinality of each cluster equals 10,000. For such a size we would use $T_2 = 10$ in the following Procedure 6.

Procedure 5 is used to search in the interval $(0, aod(S_1)]$ for the optimum grid constant. Accordingly, in Procedure 6 below, we use parameter values $a = 0$, $b = 2$, and $T_1 = 0.1$.

**Procedure 6** *Approximate Maximized Recoverable Quantity*

*Input: Two sets of 3D data points, $S_1$ and $S_2$, an interval $[a, b]$, and two thresholds $T_1$ and $T_2$ used to terminate the procedure.*

*Output: An approximate maximized recoverable quantity (i.e., subset) of $S_1$ with respect to $S_2$.*

*1. Let $a = 0$, $b = 2$, $T_1 = 0.1$, $T_2 = 10$, and compute $aod(S_1)$. Let flag = true.*

*2. While (flag = true), do the following:*

*2.1. Let grid constant $\theta = aod(S_1) \times (a + b)/2$.*

*2.2. Compute $S_1(aod(S_1))$.*

*2.3. Let $S_1(aod(S_1))$ be the input for Procedure 4; compute a set of cubes containing data points in $S_1(aod(S_1))$, denoted by $S_1'$.*

*2.4. Let $S_2$ be the input for Procedure 4; compute a set of cubes containing data points in $S_2$, denoted by $S_2'$.*

*2.5. Let $S_1'$, $S_2'$, and $\theta$ be the input for Procedure 5; compute a subset of $S_1'$, denoted by $S_1''$.*

*2.6. Let $d = d(S_1'', S_2)$.*

*2.7. If $aod(S_1'') < d$, then $b = (a + b)/2$. Otherwise $a = (a + b)/2$.*

*2.8. If $(b - a) < T_1$ or $(|S_1| - |S_1''|) < T_2$, then flag = false (i.e., break the while loop).*

*3. Output $|S_1''|$.*

Set $S_1''$ in Step 3 is called the *decided* subset of $S_1$. $S_1 \backslash S_1''$ is called the *undecided* subset of $S_1$.

Let $C_0, C_1, \ldots, C_{n-1}$ be $n$ old clusters, where $n \geq 2$. For each $i \in \{0, 1, 2, \ldots, n-1\}$; the maximized recoverable quantity of $C_i$ with respect to the union of adjacent clusters equals $i_{rq}$; and $i_n$ is the total number of adjacent clusters.

**Hypothesis 2** *We assume that the largest cardinality of a recoverable subset of old cluster $C_i$, calculated by a most powerful clustering algorithm (known until now, or to be designed by someone in the future) equals*

$$i_{rq} + \frac{|C_i| - i_{rq}}{i_n} \times 0.5$$

*if $i_n \neq 0$, or equals $|C_i|$ otherwise.*

In Hypothesis 2, we assume that a most powerful clustering algorithm can recover the whole old cluster if it is not adjacent to any other old cluster. Possibly this seems like that we overestimate the ability of the most powerful clustering algorithm. However, this is fine to estimate an upper bound for the recovery rate of any clustering algorithm. Analogously, assume that the most powerful clustering algorithm can recover the whole decided subset of size $i_{rq}$, for each old cluster. Obviously, in the case of an input data set with *strongly overlapping* (i.e., with "heavy interference" between) old clusters, the recovery rate decreases if the number of old clusters increases. Thus, for the complement of the decided subset (i.e., the undecided subset, with size $|C_i| - i_{rq}$), the recoverable quantity is divided by the number $i_n$ of adjacent clusters. The term 0.5 is based upon the consideration that there is a 50% opportunity for each data point in the undecided subset to be recovered. Such a hypothesis is straightforward if $i_n = 1$.

By Hypothesis 2 and Procedure 6, we have the following main algorithm:

**Algorithm 1** *Estimation of the Best Possible Recovery Rate*

*Input: n old clusters of 3D data points* $C_0, C_1, \ldots, C_{n-1}$.
*Output: An approximate upper bound of recovery rates of a clustering algorithm with respect to* $\cup_{i=0}^{n-1} C_i$ *as an input.*

```
1. Let $ubrr = 0$ (an initialization of the upper bound of recovery rates).
2. For each $i \in \{0, 1, 2, \ldots, n-1\}$, do the following:
    2.1. Let $i_n = 0$ (initialize the number of adjacent clusters) and $S_{i_u} = \emptyset$ (initialize the union of adjacent clusters).
    2.2. Compute $aod(C_i)$.
    2.3. For each $j \in \{0, 1, 2, \ldots, n-1\} \setminus \{i\}$, do the following:
        2.3.1. Compute $d(C_i, C_j)$.
        2.3.2. If $aod(C_i) \geq d(C_i, C_j)$, then $i_n = i_n + 1$ and $S_{i_u} = S_{i_u} \cup C_j$.
3. For each $i \in \{0, 1, 2, \ldots, n-1\}$ do the following:
    3.1. Let $C_i$ and $S_{i_u}$ as an input, apply Procedure 6 to compute an approximate maximized recoverable quantity of $C_i$ with respect to $S_{i_u}$, denoted by $i_{mrq}$.
    3.2. If $i_n \neq 0$, then (by Hypothesis 2),
             $ubrr = ubrr + i_{mrq} + \frac{|C_i| - i_{mrq}}{i_n} \times 0.5$
         Otherwise, $ubrr = ubrr + |C_i|$.
4. Output $ubrr \times 100\%$.
```

The next section contains some intermediate and final results when running Algorithm 1 on a non-trivial input data set.

## 4 Experimental Results

We use the (simulated) input data set from [2] which consists of 33 old clusters.

| $i$ | $aod(C_i)$ | $i$ | $aod(C_i)$ | $i$ | $aod(C_i)$ | $i$ | $aod(C_i)$ | $i$ | $aod(C_i)$ |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 56.28 | 7 | 81.37 | 14 | 62.92 | 21 | 61.83 | 28 | 93.45 |
| 1 | 43.38 | 8 | 66.82 | 15 | 83.53 | 22 | 46.87 | 29 | 77.16 |
| 2 | 90.23 | 9 | 74.99 | 16 | 37.37 | 23 | 49.48 | 30 | 71.19 |
| 3 | 106.92 | 10 | 20.95 | 17 | 32.90 | 24 | 59.64 | 31 | 74.70 |
| 4 | 97.52 | 11 | 110.16 | 18 | 107.16 | 25 | 77.01 | 32 | 62.06 |
| 5 | 38.94 | 12 | 78.25 | 19 | 94.86 | 26 | 40.01 | | |
| 6 | 55.16 | 13 | 68.10 | 20 | 115.19 | 27 | 103.95 | | |

**Table 1.** Averaged outlier distances in old clusters of the used input data set.

## 4.1 The Test Data Set

There are 10,000 3D data points in each cluster (which is stored in a text file named "en_angmom_f_000.i", where $i = 00, 01, 02, \ldots, 09, 10, \ldots, 32$). The union of all 33 clusters is shown in Figure 1.

Input data used in experiments always 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 $(x/20, y/11, z/11)$.

## 4.2 Some Results

Table 1 shows the averaged outlier distance of each cluster $C_i$, where $i = 0, 1, 2, \ldots, 32$. It can be computed just by applying Definition 1.

Table 2 shows cardinalities of subsets of $C_is$ induced by the corresponding averaged outlier distances $aod(C_i)$, for $i = 0, 1, 2, \ldots, 32$. Those can be computed straightforwardly, just by applying Definition 4.

We see that those induced subsets contain between 57% and 63% data points of the old clusters. This means that, when considering the question of how to select a reasonable $d$ value for creating a meaningful induced subset, the averaged outlier distance appears to be a good option for the given input data. Again, although we may replace $aod(S_1)$ in Step 2.2 in Procedure 6 by another value $d_1 \in (aod(S_1), Mod(S_1)]$ (to increase $|C_{i_{d_1}}|$, and, in consequence, to increase the recoverable quantities $i_{rq}s$ as stated in Hypothesis 2), we have to replace

| $i$ | $\lvert C_{i_{aod}}\rvert$ | $i$ | $\lvert C_{i_{aod}}\rvert$ | $i$ | $\lvert C_{i_{aod}}\rvert$ | $i$ | $\lvert C_{i_{aod}}\rvert$ | $i$ | $\lvert C_{i_{aod}}\rvert$ |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 6305 | 7 | 6113 | 14 | 5931 | 21 | 6277 | 28 | 6211 |
| 1 | 6285 | 8 | 5766 | 15 | 5921 | 22 | 6703 | 29 | 5849 |
| 2 | 6712 | 9 | 5860 | 16 | 6162 | 23 | 6305 | 30 | 5716 |
| 3 | 5786 | 10 | 6057 | 17 | 5836 | 24 | 6362 | 31 | 6279 |
| 4 | 5913 | 11 | 5744 | 18 | 6153 | 25 | 6080 | 32 | 6141 |
| 5 | 6333 | 12 | 5827 | 19 | 5908 | 26 | 6573 | | |
| 6 | 6173 | 13 | 6721 | 20 | 5872 | 27 | 5894 | | |

**Table 2.** Cardinalities of subsets of $C_is$ induced by the corresponding averaged outlier distance $aod(C_i)s$ of the input data.

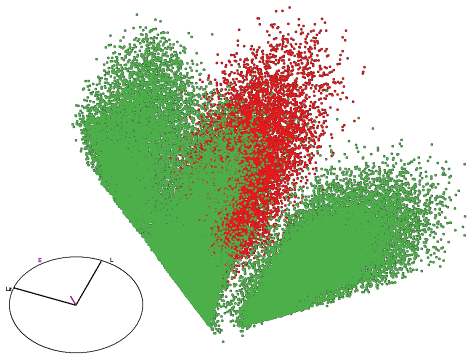

**Fig. 6.** Cluster $C_3$ with the union of 6 adjacent clusters. The upper bound of the recoverable points in $C_3$ equals 2269 (rank 11 of all 33 clusters). The experiment verified that such an upper bound seems to be a reasonable estimate.

$aod(S_1)$ in Definition 2 by $d_1$ as well. Therefore, sets $S_1$ and $S_2$ in Definition 2 are overlapping "more easily"; thus we increase the value $i_n$ in Hypothesis 2. Conversely, we replace $aod(S_1)$ in Step 2.2 in Procedure 6 by a smaller value which will decrease both $i_{rq}$ and $i_n$ in Hypothesis 2.

We applied GGobi [4] to study each cluster, together with the union of its adjacent clusters. We colored each cluster red and the union of its neighbors green. Then we rotated them in 3D such that we can view a maximum number of red data points, and capture an image for each cluster and its adjacent clusters. We call the resulting images *good* images which approximately coincide with the upper bounds of recoverable quantities. See Figures 6 and 7 for two examples.

By such good images we evaluated reasonable estimates for the upper bound of the recoverable size of each of those old clusters. See Figure 2, for example.

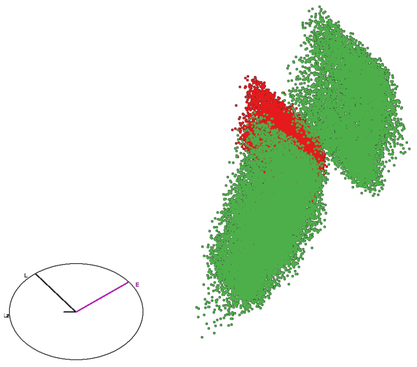

**Fig. 7.** $C_{29}$ with the union of 3 adjacent clusters. The upper bound of the recoverable points of $C_{29}$ equals 2761 (rank 12 of all 33 clusters). The experiment verified that such an upper bound seems to be a reasonable estimate.

By Definition 3, $C_{30}$ (i.e., old cluster 30) does not have any adjacent cluster (see also Table 3). So we do not have to capture a good image for $C_{30}$. See the Appendix for the remaining 31 good images.

For the example of data in [2], the approximate upper bound of recovery rates, for any clustering algorithm, is adjusted to

$$39.68\% + 166000/330000 \times 100\% = 44.71\%$$

Obviously, the provided method applies to any other clustering data set as well, and the limitation to points in 3D space is not crucial.

## 5 Conclusion

We summarize that this paper has estimated the best recovery rate of any (arbitrary) clustering algorithms with respect to any given input data set. Using the data visualization system GGobi we were also able to show that such estimates are approximately correct for the used (reasonably complex) example of a data set.

## 6 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 manuscript of this paper.

## References

1. A. B-Hur, D. Horn, H.T. Siegelmann, and V. Vapnik. Support Vector Clustering. *Journal of Machine Learning Research*, **2**:125–137, 2001.
2. Simulated astronomical data. See http://www.astro.rug.nl/ ahelmi/simulations_gaia.tar.gz
3. E. George, F. Carlos S., W. Simon D. M., and D. Marc. Gravitational clustering from scale-free initial conditions. *Monthly Notices RAS 235*, 715–748, Dec, 1988.
4. GGobi data visualization system. See http://www.ggobi.org/
5. 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.
6. A.K. Jain, M.N. Murty, and P.J. Flynn. Data clustering: A review. *ACM Computing Surveys*, 31(**3**):264–323, September 1999.
7. R. Klette and A. Rosenfeld. *Digital Geometry.* Morgan Kaufmann, San Francisco, 2004.
8. H. C. Law. Clustering, Dimensionality Reduction, and Side Information. Ph.D. Thesis, Michigan State University, the United States, 2006.
9. 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).
10. B.W. Silverman. *Density Estimation.* Chapman & Hall, London, 1986.