Applied Soft Computing 41 (2016) 390–399

Contents lists available at ScienceDirect

Applied Soft Computing

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

# Clustering by propagating probabilities between data points

Guojun Gan[a,*], Yuping Zhang[b], Dipak K. Dey[c]

[a] Department of Mathematics, Institute for Systems Genomics, Center for Health, Intervention, and Prevention (CHIP), University of Connecticut, 196 Auditorium Rd U-3009, Storrs, CT 06269, USA
[b] Department of Statistics, Institute for Systems Genomics, Center for Health, Intervention, and Prevention (CHIP), Center for Quantitative Medicine, University of Connecticut, 215 Glenbrook Road, U-4098, Storrs, CT 06269, USA
[c] Department of Statistics, University of Connecticut, 215 Glenbrook Road, U-4098, Storrs, CT 06269, USA

ARTICLE INFO

*Article history:*
Received 16 January 2015
Received in revised form
11 November 2015
Accepted 10 January 2016
Available online 28 January 2016

*Keywords:*
Affinity propagation
Data clustering
Graph-based clustering
Markov clustering
Probability propagation

ABSTRACT

In this paper, we propose a graph-based clustering algorithm called "probability propagation," which is able to identify clusters having spherical shapes as well as clusters having non-spherical shapes. Given a set of objects, the proposed algorithm uses local densities calculated from a kernel function and a bandwidth to initialize the probability of one object choosing another object as its attractor and then propagates the probabilities until the set of attractors become stable. Experiments on both synthetic data and real data show that the proposed method performs very well as expected.

## 1. Introduction

Data clustering or cluster analysis is a fundamental tool for data analysis. The goal of data clustering is to divide a set of items into groups or clusters such that items in the same cluster are more similar to each other than to items form other clusters [12,32]. As a result, data clustering has found applications in a wide range of areas such as bioinformatics [7,22], pattern recognition [20], health care [33], insurance [13,15], to just name a few.

In the past 60 years, many clustering algorithms have been developed to achieve the task of data clustering [19]. These algorithms differ significantly in terms of how clusters are defined and how the clusters are identified. The $k$-means algorithm [24] is one of the most popular and classical clustering algorithms. Used to find groups of objects with small distances among cluster members, the $k$-means algorithm starts from $k$ initial cluster centers and repeats updating cluster members and cluster centers until some stopping criterion is met. The number of clusters, $k$, is a parameter of the algorithm. One drawback of the $k$-means algorithm is that it is quite sensitive to initial cluster centers, which affect clustering results and the convergence speed. For example, [26] compared four initialization methods for the $k$-means algorithm and found that random initialization is not the best method.

To address the cluster center initialization problem, Frey and Dueck [11] proposed an efficient clustering method called affinity propagation. The Affinity Propagation (AP) algorithm starts with the similarities between pairs of data points and repeats passing real-valued messages between data points until a high-quality set of exemplars (i.e., cluster centers) and corresponding clusters are found. Unlike the $k$-means algorithm [24], the AP algorithm considers simultaneously all data points as cluster centers and thus does not suffer from the cluster center initialization problem.

One drawback of the AP algorithm is that the rules of passing messages between data points are complicated. In the AP algorithm, two types of messages are exchanged between data points: the responsibility and the availability. As we will see in Section 2, the rule for updating the responsibility involves calculating the maximum of sums of the availability and the similarity; the rule for updating the availability involves calculating the sum of positive responsibilities.

Motivated by the AP algorithm, we propose in this paper a novel clustering algorithm called "probability propagation," which is able to identify clusters having spherical shapes as well as clusters having non-spherical shapes. The probability propagation (PP) algorithm starts with a matrix of probabilities calculated from local densities and keeps propagating probabilities until the set of attractors become stable. Here we use the term "attractor" to represent a cluster center because the clusters found by the PP algorithm can have non-spherical shapes.

The PP algorithm we proposed is similar to the AP algorithm and the Markov Clustering (MCL) algorithm in that all three algorithms involve certain message-passing mechanism. One major difference between the PP algorithm and the AP algorithm is that the rules of message-passing in the former are simpler than those in the later. Another difference is that the PP algorithm is able to identify clusters of non-spherical shapes but the AP algorithm cannot. One major difference between the PP algorithm and the MCL algorithm is that the PP algorithm does not use the inflation operator, which is required by the MCL algorithm. Another difference is that the stochastic matrix initialization of the PP algorithm is different from that of the MCL algorithm.

The remaining of the paper is structured as follows. In Section 2, we give a brief description of the AP algorithm, the MCL algorithm, and spectral clustering. In Section 3, we present the PP algorithm in detail. In Section 4, we demonstrate the performance of the PP algorithm by conducting experiments on both synthetic and real data sets. In Section 5, we conclude the paper and point out some areas for future research.

* Corresponding author. Tel.: +1 860 486 3919; fax: +1 860 486 4238.
*E-mail addresses:* Guojun.Gan@uconn.edu (G. Gan), yuping.zhang@uconn.edu (Y. Zhang), dipak.dey@uconn.edu (D.K. Dey).

http://dx.doi.org/10.1016/j.asoc.2016.01.034
1568-4946/© 2016 Elsevier B.V. All rights reserved.

## 2. Literature review

In general, there are two types of clustering algorithms [21]: hierarchical and partitional. Hierarchical clustering algorithms produce a sequence of nested clusters organized as a hierarchical tree. Hierarchical clustering algorithms can be further classified into two types: agglomerative and divisive. An agglomerative algorithm starts from each object as a cluster and keeps merging clusters until all objects are in one cluster. In contrast, a divisive algorithm starts from all objects as one cluster and keeps splitting clusters until every cluster contains one object. Unlike hierarchical clustering algorithms, partitional clustering algorithms produce a single partition of the data instead of a sequence of partitions. The $k$-means algorithm, the AP algorithm, the MCL algorithm, and spectral clustering algorithms are partitional algorithms. In this section, we give a brief introduction to the AP algorithm, the MCL algorithm, and spectral clustering.

### 2.1. The AP algorithm

As we mentioned before, responsibility and availability are two types of messages exchanged between data points in the AP algorithm. The responsibility $r(i, k)$, which is sent from data point $i$ to candidate exemplar point $k$, reflects how well-suited it would be for point $k$ to be the exemplar of point $i$. The availability $a(i, k)$, which is sent from candidate exemplar point $k$ to data point $i$, reflects how appropriate it would be for data point $i$ to choose candidate exemplar $k$ as its exemplar.

The rules for updating the responsibility $r(i, k)$ and the availability $a(i, k)$ are given below [11]:

$$r(i,k) \leftarrow s(i,k) - \max_{j, j \neq k}\{a(i,j) + s(i,j)\}, \tag{1}$$

$$a(i,k) \leftarrow \min\left\{0, r(k,k) + \sum_{j, j \notin \{i,k\}} \max\{0, r(j,k)\}\right\}, \quad i \neq k, \tag{2}$$

$$a(k,k) \leftarrow \sum_{j, j \neq k} \max\{0, r(j,k)\}, \tag{3}$$

where $s(i, j)$ is the similarity between points $i$ and $j$ for $i \neq j$ and $s(k, k)$ is an input parameter called "preference." The larger the value of $s(k, k)$, the more likely that the point $k$ is to be chosen as an exemplar. To avoid numerical oscillations, the messages are damped according to a user-specified parameter.

In the AP algorithm, responsibilities and availabilities are updated according to the aforementioned rules repeatedly until some stop criterion is met. A simple stop criterion is to terminate the iterative process after a fixed number of iterations. At any step of the iterative process, responsibilities and availabilities can be combined to identify clusters and their members as follows. For data point $i$, let

$$k = \operatorname{argmax}_{j}\{a(i,j) + r(i,j)\}.$$

Then point $i$ is an exemplar or cluster center if $k = i$ and point $k$ is an exemplar for point $i$ if $k \neq i$.

### 2.2. The MCL algorithm

The MCL algorithm is a graph clustering algorithm developed by [30]. Given a data set, a graph is first created from the similarity matrix of the data set. The MCL algorithm starts from the stochastic matrix created from the graph and repeats manipulating the stochastic matrix until the stochastic matrix does not change.

Consider a data set with $n$ points. Let $G$ be a graph with $n$ vertices corresponding to the $n$ data points. Two vertices $i$ and $j$ are connected if the distance between points $i$ and $j$ is less than a threshold parameter $\delta$. The graph $G$ can be represented by an $n \times n$ matrix $T_G$ as follows: $T_G(i, j) = 1$ if $i$ and $j$ are connected or 0 if otherwise. Let $M$ be the corresponding stochastic matrix defined as:

$$M(i,j) = \frac{T_G(i,j)}{\sum_{j=1}^{n} T_G(i,j)}, \quad 1 \leq i, j \leq n.$$

The stochastic matrix is obtained by normalizing each column of the matrix $T_G$.

Once the stochastic matrix $M$ is created, the MCL algorithm proceeds to update $M$ recursively by expansion and inflation. The expansion operator $\mathrm{Exp}_t$ is defined as

$$\mathrm{Exp}_t M = M^t, \tag{4}$$

where $t$ is a positive integer. The expansion operator is responsible for allowing flow to connect different regions of the graph or network. The inflation operator $\Gamma_r$ is defined as

$$(\Gamma_r M)(i,j) = \frac{M^r(i,j)}{\sum_{j=1}^{n} M^r(i,j)}, \tag{5}$$

where $r$ is a positive real number. The inflation operator raises each element of $M$ to the $r$th power and then normalizes each column. The inflation operator is responsible for both strengthening and weakening current flow of information that influences the granularity of clusters. After a number of iterations, the matrix $M$ becomes invariant under both expansion and inflation, and all non-zero elements in every column become equal.

Clusters can be formed by observing the final stochastic matrix. Let $M^*$ be the invariant stochastic matrix obtained from the iterative process. If $M^*(i, j) = 1$, then $M^*(i, j)$ is the only non-zero entry in column $j$. In this case, points $\mathbf{x}_i$ and $\mathbf{x}_j$ are grouped to the same cluster. If $0 < M^*(i, j) < 1$, then there are $n/M^*(i, j)$ non-zero entries in column $j$. In this case, point $j$ can be grouped with any point $i$ with $M^*(i, j) > 0$. For real data sets, the later case usually does not happen. As long as the entries in a column of the stochastic matrix are different to each other, the inflation operator will reduce the small entries to zero.

The MCL algorithm requires two parameters $t$ and $r$, which are used in Eqs. (4) and (5), respectively. The default value of $t$ is 2. The parameter $r$ affects the granularity of clusters. Increasing the value of $r$ can increase the number of clusters. The default value of $r$ is also set to 2.

The MCL algorithm can be modified to handle large data sets by pruning small entries in all columns of the stochastic matrix. In the exact implementation of the MCL algorithm, the number of operations in each iteration is $O(n^3)$, where $n$ is the number of data points. If each column of the stochastic matrix is pruned to have at most $m$ non-zero entries, then the number of operations in each iterations can be reduced to $O(nm^2)$. For a more detail description of the MCL algorithm, readers are referred to [29].

### 2.3. Spectral and Kernel clustering

Although spectral clustering algorithms are not message-passing algorithms, they are graph-based algorithms. Like the MCL algorithm, spectral clustering algorithms are able to identify clusters of arbitrary shapes [23,25,27]. Spectral clustering is also related to kernel clustering. It has been pointed out that kernel $k$-means [8] and spectral clustering are two equivalent approaches [9]. In this subsection, we present a spectral clustering algorithm. In general, a spectral clustering algorithm consists of three steps [1, Chapt. 8]: first, a similarity graph of all data points is constructed; second, the data points are mapped to a feature space in which clusters are more obvious; third, a classical clustering algorithm such as $k$-means is used to partition the transformed data points. In the first step, there are several ways to construct a similarity graph [1, Chap. 8]: the $k$-nearest neighbor method, the $\epsilon$-neighborhood method, and the fully connected graph. The second step is referred to as spectral embedding and is achieved by the eigenvectors of the graph Laplacian [1, Chap. 8].

Let $X = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$ be a dataset with $n$ points. Let $W$ be a nonnegative weighted $n \times n$ adjacency matrix constructed from $X$. Under the $\epsilon$-neighbor method, for example, $W_{ij} = W_{ji} = 1$ if $||\mathbf{x}_i - \mathbf{x}_j||^2 < \epsilon$ and $W_{ij} = W_{ji} = 0$ if otherwise. To create a fully connected graph, we can use $W_{ij} = \exp(-d_{ij}^2/\sigma^2)$, where $d_{ij}$ is the distance between points $\mathbf{x}_i$ and $\mathbf{x}_j$ and $\delta$ is a bandwidth. The degree matrix $D$ is the diagonal matrix such that

$$D_{ii} = \sum_{j=1}^{n} W_{ij}, \quad i = 1, 2, \ldots, n.$$

The unnormalized graph Laplacian $L$ is defined as

$$L = D - W.$$

The normalized graph Laplacian can be defined as

$$L_{sym} = D^{-1/2} L D^{-1/2} = I - D^{-1/2} W D^{-1/2}$$

or

$$L_{rw} = D^{-1} L = I - D^{-1} W.$$

Depending on whether the graph Laplacian is normalized or not, spectral clustering algorithms can be classified into two groups [23]: unnormalized spectral clustering and normalized spectral clustering. As pointed out in [23], normalized spectral clustering with the normalized graph Laplacian $L_{rw}$ is preferable. The major steps of such a normalized spectral clustering algorithm is given below [1, Chap. 8]:

1. Construct the adjacency matrix $W$ and the degree matrix $D$;
2. Calculate the normalized graph Laplacian $L_{rw}$;
3. Compute the top $H$ eigenvectors $\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_H$ (i.e., the eigenvectors corresponding to the $H$ smallest eigenvalues) of $L_{rw}$;
4. Construct the $n \times H$ matrix $U$ with $\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_H$ of $L_{rw}$ as columns;
5. Normalize the rows of $U$ such that
$$\sum_{j=1}^{H} U_{ij}^2 = 1, \quad i = 1, 2, \ldots, n;$$
6. Apply the $k$-means algorithm to the rows of the matrix $U$.

## 3. The PP algorithm

The PP algorithm consists of two components: creating a stochastic matrix (i.e., a matrix of probabilities) from a data set based on a bandwidth parameter supplied by users and partitioning the data set by propagating the probabilities. In this section, we present the PP algorithm.

Let $X = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$ be a data set of $n$ points. Let $W$ be an $n \times n$ matrix such that the entry $W(i, j)$ represents the probability that point $\mathbf{x}_i$ chooses point $\mathbf{x}_j$ as its attractor. Here we use an attractor to represent a "cluster center". Intuitively, an attractor is a "center" for an arbitrarily shaped cluster. Since a point can choose one of the $n$ points including itself as its attractor, we have

$$\sum_{j=1}^{n} W(i, j) = 1, \quad 1 \le i \le n.$$

Hence $W$ is a stochastic matrix.

At the beginning of the PP algorithm, the stochastic matrix $W$ is initialized in such a way that the probability that a point chooses a neighbor as its attractor is proportional to the local density around the neighbor. Let $\delta$ be a bandwidth parameter and let $D(\cdot, \cdot)$ be a distance function. The local density around point $\mathbf{y}$ is estimated as

$$V(\mathbf{y}) = \sum_{\mathbf{x} \in N(\mathbf{y})} K\left(\frac{D(\mathbf{x}, \mathbf{y})}{\delta}\right), \tag{6}$$

where $K$ is a kernel function and

$$N(\mathbf{y}) = \{\mathbf{x} \in X : D(\mathbf{x}, \mathbf{y}) < \delta\}. \tag{7}$$

We first create a matrix $W_0$ as follows:

$$W_0(i, j) = \begin{cases} V(\mathbf{x}_j), & \text{if } \mathbf{x}_j \in N(\mathbf{x}_i), \\ 0, & \text{if } \mathbf{x}_j \notin N(\mathbf{x}_i), \end{cases} \quad 1 \le i, j \le n. \tag{8}$$

For each $1 \le i \le n$, we sort $W_0(i, 1), W_0(i, 2), \ldots, W_0(i, n)$ such that

$$W_0(i, i_1) \ge W_0(i, i_2) \ge \cdots \ge W_0(i, i_n),$$

where $(i_1, i_2, \ldots, i_n)$ is a permutation of $(1, 2, \ldots, n)$. Then we create the initial stochastic matrix as follows:

$$W(i, i_j) = \begin{cases} \dfrac{W_0(i, i_j)}{\sum_{r=1}^{s} W_0(i, i_r)}, & \text{if } 1 \le j \le s \\ 0, & \text{if } s+1 \le j \le n. \end{cases}, \quad i = 1, 2, \ldots, n, \tag{9}$$

where $1 \le s \le n$ is a parameter used to control the shape of the clusters. By the above definition, we have

$$\sum_{j=1}^{n} W(i, j) = \sum_{r=1}^{s} \frac{W_0(i, i_r)}{\sum_{l=1}^{s} W_0(i, i_l)} = 1, \quad 1 \le i \le n,$$

which shows that $W$ is a stochastic matrix. Based on this definition, each data point can choose at most $s$ data points as its initial attractors with positive probabilities. The distances between point $\mathbf{x}_i$ and its candidate attractors are less than $\delta$. In addition, the candidate attractors of point $\mathbf{x}_i$ have higher densities than other points in $N(\mathbf{x}_i)$ if $N(\mathbf{x}_i)$ contains more than $s$ points.

A very general approach for the choice of $D(\cdot, \cdot)$ is the Bregman divergence [3,4], which includes squared Euclidean, Kullback 'Leibler divergence, Itakura' Saito distance, and Mahalanobis distance as special cases. The bandwidth selection issues will be discussed next.

There are several choices for the kernel function [28, p. 43]. For example, we can use the uniform kernel, the Gaussian kernel, or the triangle kernel, which are defined as

$$K_U(u) = \begin{cases} \frac{1}{2}, & \text{if } |u| \le 1, \\ 0, & \text{if } |u| > 1, \end{cases} \tag{10}$$

$$K_G(u) = \frac{1}{\sqrt{2\pi}} e^{-1/2u^2}, \tag{11}$$

$$K_T(u) = \begin{cases} 1 - |u|, & \text{if } |u| \le 1, \\ 0, & \text{if } |u| > 1, \end{cases} \tag{12}$$

respectively. Fig. 1 shows the shapes of the three kernels. The uniform kernel gives equal weights to all points in the interval; whereas the triangle kernel gives the most weights to the points in the center. The Gaussian kernel is something in between.

Once the stochastic matrix is initialized, the PP algorithm repeats propagating probabilities between data points until some

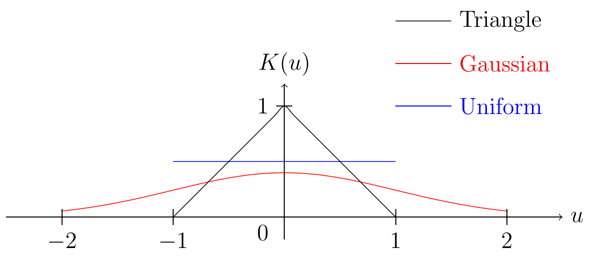


**Fig. 1.** Shapes of the uniform, Gaussian, and triangle kernels.

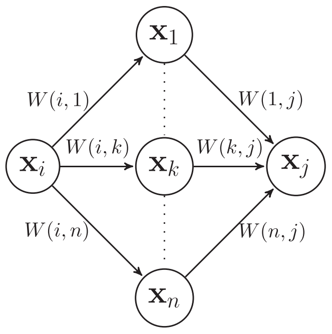


**Fig. 2.** Probability propagation between two data points. Here $W(i, k)$ denotes the probability of point $\mathbf{x}_k$ being chosen as the attractor of point $\mathbf{x}_i$. Similarly, $W(k, j)$ denotes the probability of point $\mathbf{x}_j$ being chosen as the attractor of point $\mathbf{x}_k$.

stopping criterion is met. The probabilities are propagated as follows

$$W(i,j) \leftarrow \sum_{k=1}^{n} W(i,k)W(k,j), \quad 1 \le i,j \le n.$$

Fig. 2 shows how probabilities are updated between two data points. In matrix form, the probability propagation can be written as

$$W \leftarrow W^2. \tag{13}$$

After $m$ iterations, the stochastic matrix becomes $W^{2^m}$, indicating that the stochastic matrix evolves exponentially.

Algorithm 1 shows the pseudo-code of the PP algorithm. The algorithm has two parameters: $\delta$ and $s$. The first parameter is the bandwidth used to define neighbors. The second parameter is used to control the shape of the clusters. In general, we use a large $s$ (e.g., $s = n$) if we want to find arbitrarily-shaped clusters and use a small $s$ (e.g., $1 \le s \le 10$) if we want to find spherical clusters. The algorithm terminates if the set of attractors does not change in two consecutive iterations.

**Algorithm 1** *(Pseudo-code of the PP algorithm).*

```
Require: A data set, $\delta$, $s$
  Calculate the distance matrix of the data set
  Initialize the stochastic matrix $W$ according to Eq. (9)
  while true do
    $W \leftarrow W^2$
    if The set of attractors did not change then
      break
    end if
  end while
  Find the attractors and their corresponding clusters
```

Once the iterative process terminates, the set of attractors can be identified from the final stochastic matrix. Let $W^*$ be the stable stochastic matrix. For every $i = 1, 2, \ldots, n$, let $k_i$ be defined as

$$k_i = arg \max_{1 \le j \le n} W(i,j).$$

The point $\mathbf{x}_{k_i}$ is then attractor for point $\mathbf{x}_i$. Different points can choose the same point as their attractor. The number of different attractors is the number of clusters.

Since the stochastic matrix evolves exponentially, the PP algorithm usually converges in several iterations. Fig. 3(a) shows a 2-dimensional data set with 40 points. Fig. 4 shows how the set of attractors changes from initialization to convergence. The PP algorithm converges in a few iterations.

With fixed $s$, the bandwidth controls the number of clusters. In general, increasing the bandwidth decreases the number of clusters. Fig. 3(b) shows the relationship between the bandwidth and the number of clusters. One approach to choosing a value for the bandwidth is to look at the distances between $n(n-1)/2$ pairs of data points. For example, we can try some percentiles of these distances. In general, using the 50th percentile or larger numbers as the bandwidth will group the whole data set into a single cluster. Like the MCL algorithm, the number of operations in each iteration is $O(n^3)$ because each iteration involves a matrix multiplication.

## 4. Experiments

In this section, we present some numerical experiments based on both synthetic and real data sets to demonstrate the performance of the PP algorithm.

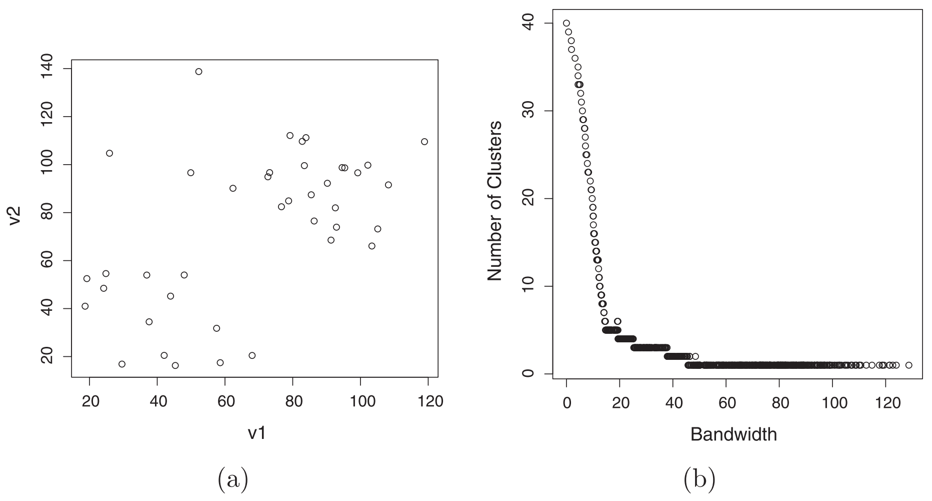


**Fig. 3.** The left figure shows a data set with 40 points. The right figure shows the number of clusters obtained by applying the PP algorithm to the 40-point data set with $s = 40$ and different bandwidths.

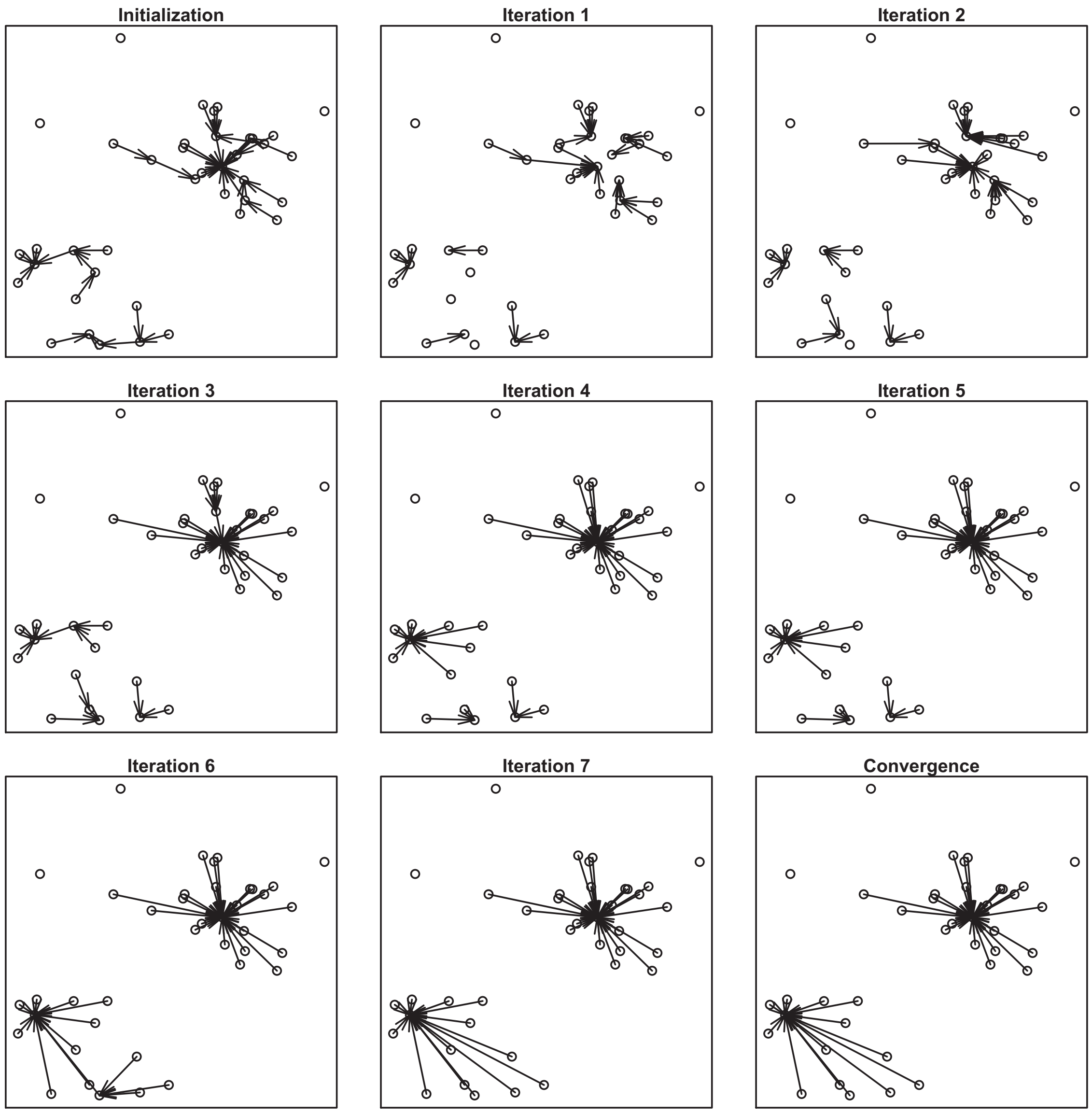


**Fig. 4.** Application of the PP algorithm to the data set with 40 points. The Gaussian kernel was used and the bandwidth value $\delta$ was set to be 16.27. An arrow pointing from point $A$ to point $B$ means that point $A$ chooses point $B$ as its attractor. A point without arrow means that it chooses itself as its attractor.

### 4.1. Evaluation method

To measure the accuracy of the clustering results, we use the corrected Rand index [7]. The value of the corrected Rand index ranges from −1 to 1. A corrected Rand index of 1 indicates a perfect agreement between the known partition and the found partition; whereas a negative corrected Rand index indicates agreement by chance. One advantage of the corrected Rand index is that it is not biased toward a given clustering algorithm or the number of clusters found by the algorithm.

To define the corrected Rand index mathematically, we let $\{U_1, U_2, \ldots, U_{k_1}\}$ be the known partition and let $\{V_1, V_2, \ldots, V_{k_2}\}$ be the partition found by a clustering algorithm. Then the corrected Rand index is defined as

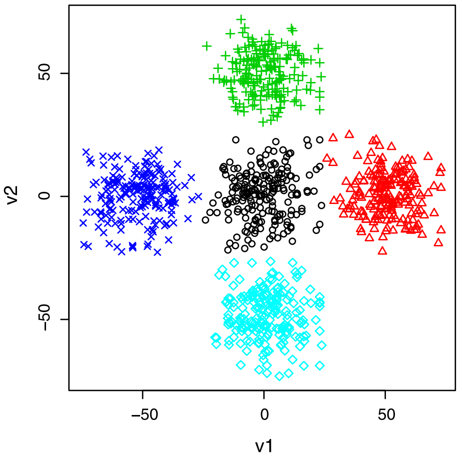
$$R = \frac{\sum_{i=1}^{k_1}\sum_{j=1}^{k_2}\binom{n_{ij}}{2} - \binom{n}{2}^{-1}\sum_{i=1}^{k_1}\binom{n_{i\cdot}}{2}\sum_{j=1}^{k_2}\binom{n_{\cdot j}}{2}}{\frac{1}{2}\left[\sum_{i=1}^{k_1}\binom{n_{i\cdot}}{2} + \sum_{j=1}^{k_2}\binom{n_{\cdot j}}{2}\right] - \binom{n}{2}^{-1}\sum_{i=1}^{k_1}\binom{n_{i\cdot}}{2}\sum_{j=1}^{k_2}\binom{n_{\cdot j}}{2}}, \tag{14}$$

where $n_{ij} = |U_i \cap V_j|$, $n_{i\cdot} = |U_i|$, $n_{\cdot j} = |V_j|$, $n$ is the total number of data points, and $|\cdot|$ indicates cardinality.

### 4.2. Experiments on synthetic data

To test the performance of the PP algorithm on synthetic data, we created two synthetic data sets (see Fig. 5). The first synthetic data set, shown in Fig. 5(a), has five spherical clusters, each of which contains 200 data points. The second synthetic data set contains two non-spherical clusters as shown in Fig. 5(b), from which we see that the data points are contained in two nested circles.

We applied the PP algorithm to the first synthetic data set with different parameters. Table 1 summarizes the clustering results obtained by the PP algorithm. In these runs, we used the 2th, 6th,


(a)

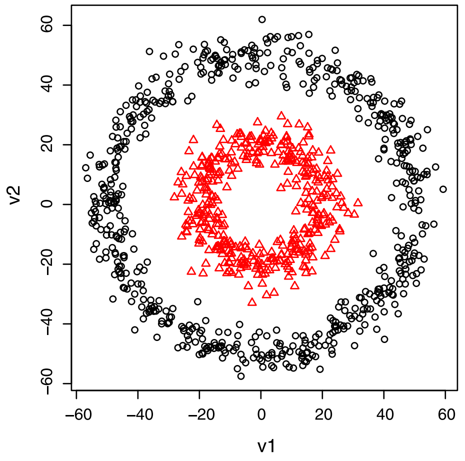

(b)

**Fig. 5.** Two synthetic data sets. (a) The first synthetic data set with five spherical clusters. (b) The second synthetic data set with two non-spherical clusters.

**Table 1**
The accuracy and speed of the PP algorithm when applied to the first synthetic data set with the uniform kernel and various parameters. Columns *K*, *R*, and *T* denote the number of found clusters, the corrected Rand index, and the runtime in seconds.

| $\delta$ | $s=1$ | | | $s=100$ | | | $s=1000$ | | |
|---|---|---|---|---|---|---|---|---|---|
| | *K* | *R* | *T* | *K* | *R* | *T* | *K* | *R* | *T* |
| 6.3 | 21 | 0.83 | 2.66 | 16 | 0.97 | 7.83 | 16 | 0.97 | 7.96 |
| 11.61 | 5 | 1 | 1.8 | 5 | 0.99 | 3.21 | 5 | 0.99 | 3.69 |
| 16.15 | 5 | 1 | 1.85 | 5 | 1 | 2.72 | 5 | 1 | 2.56 |
| 30.38 | 3 | 0.48 | 1.77 | 1 | 0 | 4.23 | 1 | 0 | 5.33 |

10th and 20th percentiles of all distances for the parameter $\delta$ and used three different values for the parameter *s*. From Table 1 we see that for fixed *s*, a larger value for $\delta$ leads to less number of clusters. The reason is that if a large $\delta$ is used, then a small number of attractors with high local densities will be selected by other points. From Table 1 we also see that for fixed $\delta$, a larger value for *s* leads to less number of clusters. This is the case because when a larger *s* is used, a point gets more connections, which help the point connect to an attractor with high local densities.

The clustering results shown in Table 1 were obtained by the PP algorithm with the uniform kernel. We also applied the PP algorithm to the first synthetic data with the Gaussian kernel and the triangle kernel. The results are shown in Tables 2 and 3. We used the same combinations of parameters for $\delta$ and *s* but used different kernels to initialize the stochastic matrix. Comparing Tables 1 and 2, we see that the uniform kernel and the Gaussian kernel have little impact on the accuracy. However, the PP algorithm with the Gaussian kernel is a little bit faster than the algorithm with the uniform kernel, especially when *s* is large.

Comparing Tables 1–3, we can see that the PP algorithm with the triangle kernel performs the best in terms of overall accuracy and speed. The algorithm with the triangle kernel is less sensitive to the parameter $\delta$. Consider $s=1$, for example, the PP algorithm with the triangle kernel still produced five clusters when $\delta=30.38$.

We also applied the AP algorithm, the MCL algorithm, and the normalized spectral clustering algorithm to the first synthetic data set with various parameters. The clustering results obtained by the three algorithms are summarized in Tables 4–6. The AP algorithm has two parameters: the preference value and the damping factor. The MCL algorithm has three parameters: the threshold $\delta$ used to construct the graph, the expansion parameter *t*, and the infla-

**Table 2**
The accuracy and speed of the PP algorithm when applied to the first synthetic data set with the Gaussian kernel and various parameters. Columns *K*, *R*, and *T* denote the number of found clusters, the corrected Rand index, and the runtime in seconds.

| $\delta$ | $s=1$ | | | $s=100$ | | | $s=1000$ | | |
|---|---|---|---|---|---|---|---|---|---|
| | *K* | *R* | *T* | *K* | *R* | *T* | *K* | *R* | *T* |
| 6.3 | 19 | 0.91 | 2.61 | 16 | 0.97 | 4.8 | 16 | 0.97 | 4.87 |
| 11.61 | 5 | 1 | 1.79 | 5 | 0.99 | 3.38 | 5 | 0.99 | 3.5 |
| 16.15 | 5 | 1 | 1.91 | 5 | 1 | 2.79 | 5 | 1 | 2.89 |
| 30.38 | 2 | 0.23 | 1.71 | 1 | 0 | 4.64 | 1 | 0 | 5.73 |

**Table 3**
The accuracy and speed of the PP algorithm when applied to the first synthetic data set with the triangle kernel and various parameters. Columns *K*, *R*, and *T* denote the number of found clusters, the corrected Rand index, and the runtime in seconds.

| $\delta$ | $s=1$ | | | $s=100$ | | | $s=1000$ | | |
|---|---|---|---|---|---|---|---|---|---|
| | *K* | *R* | *T* | *K* | *R* | *T* | *K* | *R* | *T* |
| 6.3 | 30 | 0.72 | 3.02 | 16 | 0.97 | 5.66 | 16 | 0.97 | 5.59 |
| 11.61 | 5 | 0.99 | 2.61 | 5 | 0.99 | 3.62 | 5 | 0.99 | 3.67 |
| 16.15 | 5 | 1 | 2.01 | 5 | 1 | 2.86 | 5 | 1 | 2.92 |
| 30.38 | 5 | 0.92 | 1.42 | 5 | 0.99 | 2.11 | 1 | 0 | 6.31 |

**Table 4**
The accuracy and speed of the AP algorithm when applied to the first synthetic data set with different preference values. The preferences values correspond to $-3^i$ for $i=0, 1, ldots, 9$. In these runs, the damping factor was set to 0.9. In addition, *K*, *R*, and *T* denote the number of found clusters, the corrected Rand index, and the runtime in seconds.

| Preference | *K* | *R* | *T* | Preference | *K* | *R* | *T* |
|---|---|---|---|---|---|---|---|
| −1 | 863 | 0 | 256.9 | −243 | 76 | 0.11 | 65.64 |
| −3 | 611 | 0.01 | 270.8 | −729 | 44 | 0.19 | 50.56 |
| −9 | 377 | 0.02 | 242.32 | −2187 | 24 | 0.32 | 51.93 |
| −27 | 225 | 0.04 | 101.46 | −6561 | 15 | 0.46 | 48.48 |
| −81 | 132 | 0.07 | 48.42 | −19683 | 5 | 1 | 32.78 |

**Table 5**
The accuracy and speed of the MCL algorithm when applied to the first synthetic data set with various parameters. The expansion parameter used in these runs was set to 2. In addition, *K*, *R*, and *T* denote the number of found clusters, the corrected Rand index, and the runtime in seconds.

| $\delta$ | $r=0.5$ | | | $r=1$ | | | $r=4$ | | |
|---|---|---|---|---|---|---|---|---|---|
| | *K* | *R* | *T* | *K* | *R* | *T* | *K* | *R* | *T* |
| 6.3 | 15 | 0.76 | 8.53 | 15 | 0.76 | 37.73 | 234 | 0.06 | 21.58 |
| 11.61 | 1 | 0 | 7.54 | 5 | 0.99 | 10.68 | 58 | 0.29 | 14.71 |
| 16.15 | 1 | 0 | 6.61 | 5 | 1 | 7.33 | 15 | 0.97 | 14.28 |
| 30.38 | 1 | 0 | 5.96 | 1 | 0 | 8.79 | 5 | 1 | 7.14 |

**Table 6**
The accuracy and speed of the normalized spectral clustering algorithm when applied to the first synthetic data set with various parameters. In addition, $K$, $R$, and $T$ denote the number of found clusters, the corrected Rand index, and the runtime in seconds.

| $\epsilon$ | $H=1$ | | | $H=5$ | | | $H=10$ | | |
|---|---|---|---|---|---|---|---|---|---|
| | $K$ | $R$ | $T$ | $K$ | $R$ | $T$ | $K$ | $R$ | $T$ |
| 6.3 | 5 | 0 | 43.69 | 5 | 0.75 | 40.45 | 5 | 0.75 | 40.19 |
| 11.61 | 5 | 0 | 42.59 | 5 | 0.99 | 47.6 | 5 | 0.99 | 42.48 |
| 16.15 | 5 | 0 | 53.23 | 5 | 1 | 41.34 | 5 | 0.99 | 41.41 |
| 30.38 | 5 | 0 | 30.45 | 5 | 1 | 30.07 | 5 | 0.98 | 30.44 |

**Table 9**
The accuracy and speed of the MCL algorithm when applied to the second synthetic data set with various parameters. The expansion parameter used in these runs was set to 2. In addition, $K$, $R$, and $T$ denote the number of found clusters, the corrected Rand index, and the runtime in seconds.

| $\delta$ | $r=0.5$ | | | $r=1$ | | | $r=4$ | | |
|---|---|---|---|---|---|---|---|---|---|
| | $K$ | $R$ | $T$ | $K$ | $R$ | $T$ | $K$ | $R$ | $T$ |
| 6.52 | 4 | 0.995 | 8.33 | 5 | 0.882 | 33.12 | 150 | 0.022 | 18.48 |
| 13.66 | 1 | 0 | 7.24 | 2 | 1 | 12.55 | 37 | 0.064 | 14.3 |
| 20.53 | 1 | 0 | 6.05 | 1 | 0 | 8.87 | 25 | 0.09 | 13.01 |
| 30.75 | 1 | 0 | 4.98 | 1 | 0 | 5.67 | 15 | 0.047 | 13.83 |

**Table 10**
The accuracy and speed of the normalized spectral clustering algorithm when applied to the second synthetic data set with various parameters. In addition, $K$, $R$, and $T$ denote the number of found clusters, the corrected Rand index, and the runtime in seconds.

| $\epsilon$ | $H=1$ | | | $H=5$ | | | $H=10$ | | |
|---|---|---|---|---|---|---|---|---|---|
| | $K$ | $R$ | $T$ | $K$ | $R$ | $T$ | $K$ | $R$ | $T$ |
| 6.52 | 2 | 1 | 34.38 | 2 | 1 | 43.47 | 2 | 1 | 30.17 |
| 13.66 | 2 | 0 | 39.62 | 2 | 1 | 46.98 | 2 | 0.17 | 40.14 |
| 20.53 | 2 | 0 | 45.15 | 2 | 1 | 42.3 | 2 | 0.01 | 36.25 |
| 30.75 | 2 | 0 | 43.34 | 2 | 0.93 | 42.56 | 2 | 0.07 | 38.77 |

**Table 11**
Four real data sets. The C-Cube data set contains the first 2000 records from the C-Cube test set, which contains 19133 records.

| Dataset | Samples | Attributes | Cluster sizes |
|---|---|---|---|
| Alizadeh-V2 [2] | 62 | 2093 | 42, 9, 11 |
| Gordon [18] | 181 | 1626 | 31, 150 |
| Multiple features [10] | 2000 | 649 | 10 equal-sized clusters |
| C-Cube [6,5] | 2000 | 34 | 21 clusters |

tion parameter $r$. In the AP and MCL algorithms, we use the same stop criterion, which is to terminate the iterative process when the cluster members do not change in two consecutive iterations. The normalized spectral clustering algorithm has three parameters: the threshold $\epsilon$ used to construct the adjacency matrix, the number $H$ of eigenvectors, and the number $K$ of clusters. Comparing Tables 3 and 6, we see that the normalized spectral clustering algorithm was slower than the PP algorithm when applied to the first synthetic data set.

From Table 4 we see that the AP algorithm requires a large preference value (in magnitude) in order to recover the clusters correctly. If we compare the runtime in Tables 3 and 4, the AP algorithm converged slower than the PP algorithm, especially when small preference values (in magnitude) were used. From Table 5 we see that the MCL algorithm requires a large inflation parameter in order to recover the clusters correctly. In addition, the MCL algorithm converged slower than the PP algorithm but faster than the AP algorithm.

We applied the PP algorithm with the triangle kernel to the second synthetic data set with various parameters. The results are summarized in Table 7. Since this data set has non-spherical clusters, using $s=1$ in the PP algorithm is not able to recover the non-spherical clusters correctly. The PP algorithm requires a large value of $s$ in order to recover such non-spherical clusters correctly. Since the two clusters are nested, the PP algorithm with large values of $\delta$ will cluster the whole data set into one cluster.

We also applied the AP algorithm, the MCL algorithm, and the normalized spectral clustering algorithm to the second synthetic data set with various parameters. The results obtained by the two algorithms are summarized in Tables 8–10, respectively.

**Table 7**
The accuracy and speed of the PP algorithm when applied to the second synthetic data set with the triangle kernel and various parameters. Columns $K$, $R$, and $T$ denote the number of found clusters, the corrected Rand index, and the runtime in seconds.

| $\delta$ | $s=1$ | | | $s=100$ | | | $s=1000$ | | |
|---|---|---|---|---|---|---|---|---|---|
| | $K$ | $R$ | $T$ | $K$ | $R$ | $T$ | $K$ | $R$ | $T$ |
| 6.52 | 27 | 0.08 | 2.6 | 4 | 0.995 | 9.67 | 4 | 0.995 | 8.76 |
| 13.66 | 9 | 0.26 | 1.8 | 1 | 0 | 7.79 | 1 | 0 | 7.91 |
| 20.53 | 7 | 0.2 | 1.84 | 1 | 0 | 4.51 | 1 | 0 | 4.77 |
| 30.75 | 1 | 0 | 1.72 | 1 | 0 | 2.59 | 1 | 0 | 2.81 |

**Table 8**
The accuracy and speed of the AP algorithm when applied to the second synthetic data set with different preference values. The preferences values correspond to $-3^i$ for $i=0, 1, \ldots, 9$. In these runs, the damping factor was set to 0.9. In addition, $K$, $R$, and $T$ denote the number of found clusters, the corrected Rand index, and the runtime in seconds.

| Preference | $K$ | $R$ | $T$ | Preference | $K$ | $R$ | $T$ |
|---|---|---|---|---|---|---|---|
| −1 | 868 | 0.001 | 263.49 | −243 | 65 | 0.03 | 52.44 |
| −3 | 602 | 0.002 | 255.48 | −729 | 33 | 0.058 | 44.95 |
| −9 | 356 | 0.005 | 275.83 | −2187 | 26 | 0.075 | 49.18 |
| −27 | 202 | 0.01 | 97.6 | −6561 | 17 | 0.112 | 41.4 |
| −81 | 123 | 0.017 | 66.14 | −19683 | 12 | 0.158 | 39.53 |

From Table 8 we see that the AP algorithm was not able to recover the two non-spherical clusters correctly. Table 9 shows that the MCL algorithm was able to recover the non-spherical clusters with appropriate parameter values. Comparing Tables 7–10, we see that the PP algorithm converged faster than the other three algorithms.

## 4.3. Experiments on real data

To test the PP algorithm on real data, we obtain two gene expression data sets from [7][1]: the gene expression data from blood cancers and the gene expression data from lung cancers. Both data sets have known labels. We also obtained one data set from the UCI machine Learning Repository [10] and a subset of the C-Cube data set [5,6]. Table 11 shows the information of the four real data sets. For the two gene expression data sets, we use Spearman's rank correlation coefficient to measure the similarity between data points. In other words, we use the following distance measure:

$$d(\mathbf{x}, \mathbf{y}) = 1 - \rho(\mathbf{x}, \mathbf{y}),$$

where $\rho(\cdot, \cdot)$ is Spearman's rank correlation coefficient [31, p. 505]. The third and the fourth data sets consist of features of handwritten digits and characters, respectively.

We applied the PP algorithm, the AP algorithm, the MCL algorithm, and the normalized spectral clustering algorithm to the Alizadeh-V2 data set with different parameters. For the parameter $\delta$ used in the PP, MCL, and spectral algorithms, in particular, we used the 2th, 6th, 10th, and 20th percentiles of the all distances calculated from the data set. In the AP algorithm, we used $-2^i$ for

[1] The two data sets are available at http://bioinformatics.rutgers.edu/Static/Supplements/CompCancer/datasets.htm.

$i=0, 1, \ldots, 9$ as preference values and a damping factor of 0.9. The clustering results are summarized in Tables 12–15, respectively. From these tables we see that the runtimes of the three algorithms are approximately the same. Since the Alizadeh-V2 data set contains only 62 data points, all the three algorithms converged in about 1.5 s. In terms of accuracy, the PP algorithm and the MCL algorithm produced more accurate results than the AP algorithm did. In addition, the PP algorithm is less sensitive to its parameters than the MCL algorithm. For example, the PP algorithm with different values of $s$ produced relatively same results. However, the MCL algorithm with different values of $r$ produced very different results. Comparing Tables 12 and 15 we see that the PP algorithm was able to produce more accurate results than the normalized spectral clustering algorithm.

For the Gordon data set, the clustering results obtained by the three algorithms are summarized in Tables 16–19, respectively. Since this data set contains only 181 data points, all the three algorithms finished partitioning the data within approximately the same amount of time. In terms of accuracy, the PP algorithm produced more accurate than AP algorithm did. In addition, the PP algorithm is less sensitive to the parameter $s$ than the MCL algorithm is to the parameter $r$. From Table 19, we see that the normalized spectral clustering algorithm failed to produce meaningful results for the Gordon data set.

We applied the PP algorithm, the AP algorithm, the MCL algorithm, and the normalized spectral clustering algorithm to the multiple feature data set with different parameters. For this data set, the AP algorithm did not converge. The results of the other three algorithms are summarized in Tables 20–22, respectively. From Table 20 we see that the PP algorithm was able to produce meaningful partitions with $s=1$ and small bandwidths. Table 21 shows that the MCL algorithm assigned all points into one clusters for all the cases. Table 22 shows that the normalized spectral clustering algorithm was able to produce meaningful results when $H$ was set appropriately. In terms of running time, the PP algorithm is the

**Table 12**
The accuracy and speed of the PP algorithm when applied to the Alizadeh-V2 data set with the triangle kernel and various parameters.

| $\delta$ | $s=1$ | | | $s=7$ | | | $s=62$ | | |
|---|---|---|---|---|---|---|---|---|---|
| | $K$ | $R$ | $T$ | $K$ | $R$ | $T$ | $K$ | $R$ | $T$ |
| 0.4 | 49 | 0.049 | 1.48 | 49 | 0.049 | 1.47 | 49 | 0.049 | 1.53 |
| 0.59 | 29 | 0.031 | 1.3 | 29 | 0.031 | 1.33 | 29 | 0.031 | 1.33 |
| 0.67 | 13 | 0.173 | 1.28 | 10 | 0.523 | 1.3 | 10 | 0.523 | 1.3 |
| 0.79 | 3 | 0.727 | 1.27 | 3 | 0.788 | 1.29 | 3 | 0.788 | 1.34 |

**Table 13**
The accuracy and speed of the AP algorithm when applied to the Alizadeh-V2 data set with different preference values.

| Preference | $K$ | $R$ | $T$ | Preference | $K$ | $R$ | $T$ |
|---|---|---|---|---|---|---|---|
| −1 | 9 | 0.197 | 2.09 | −32 | 1 | 0 | 1.5 |
| −2 | 4 | 0.464 | 1.58 | −64 | 1 | 0 | 1.47 |
| −4 | 2 | 0.708 | 1.44 | −128 | 1 | 0 | 1.51 |
| −8 | 2 | 0.708 | 1.47 | −256 | 1 | 0 | 1.45 |
| −16 | 2 | 0.708 | 1.55 | −512 | 1 | 0 | 1.4 |

**Table 14**
The accuracy and speed of the MCL algorithm when applied to the Alizadeh-V2 data set with various parameters. The expansion parameter used in these runs was set to 2.

| $\delta$ | $r=0.5$ | | | $r=1$ | | | $r=4$ | | |
|---|---|---|---|---|---|---|---|---|---|
| | $K$ | $R$ | $T$ | $K$ | $R$ | $T$ | $K$ | $R$ | $T$ |
| 0.4 | 49 | 0.049 | 1.36 | 50 | 0.043 | 1.37 | 50 | 0.043 | 1.36 |
| 0.59 | 29 | 0.031 | 1.4 | 29 | 0.031 | 1.36 | 30 | 0.113 | 1.29 |
| 0.67 | 10 | 0.523 | 1.39 | 11 | 0.542 | 1.3 | 25 | 0.078 | 1.3 |
| 0.79 | 2 | -0.022 | 1.4 | 3 | 0.788 | 1.3 | 14 | 0.065 | 1.3 |

**Table 15**
The accuracy and speed of the normalized spectral clustering algorithm when applied to the Alizadeh-V2 data set with various parameters. In these runs, fully connected graphs were used.

| $\delta$ | $H=1$ | | | $H=5$ | | | $H=10$ | | |
|---|---|---|---|---|---|---|---|---|---|
| | $K$ | $R$ | $T$ | $K$ | $R$ | $T$ | $K$ | $R$ | $T$ |
| 0.4 | 3 | 0 | 2.17 | 3 | 0.39 | 1.73 | 3 | 0.38 | 1.7 |
| 0.59 | 3 | 0 | 2.09 | 3 | 0.42 | 1.67 | 3 | 0.57 | 1.97 |
| 0.67 | 3 | 0 | 2.15 | 3 | 0.39 | 1.62 | 3 | 0.54 | 2.1 |
| 0.79 | 3 | 0 | 1.67 | 3 | 0.4 | 1.75 | 3 | 0.33 | 1.98 |

**Table 16**
The accuracy and speed of the PP algorithm when applied to the Gordon data set with the triangle kernel and various parameters.

| $\delta$ | $s=1$ | | | $s=19$ | | | $s=181$ | | |
|---|---|---|---|---|---|---|---|---|---|
| | $K$ | $R$ | $T$ | $K$ | $R$ | $T$ | $K$ | $R$ | $T$ |
| 0.49 | 58 | 0.071 | 8.53 | 48 | 0.334 | 8.74 | 48 | 0.334 | 8.47 |
| 0.54 | 16 | 0.588 | 8.52 | 13 | 0.826 | 9.85 | 13 | 0.826 | 9.06 |
| 0.57 | 8 | 0.875 | 8.65 | 8 | 0.875 | 9.52 | 8 | 0.875 | 8.99 |
| 0.61 | 3 | 0.925 | 8.65 | 3 | 0.951 | 9.45 | 3 | 0.951 | 9.05 |

**Table 17**
The accuracy and speed of the AP algorithm when applied to the Gordon data set with different preference values.

| Preference | $K$ | $R$ | $T$ | Preference | $K$ | $R$ | $T$ |
|---|---|---|---|---|---|---|---|
| −1 | 15 | 0.059 | 11.06 | −32 | 1 | 0 | 10.12 |
| −2 | 5 | 0.168 | 10.72 | −64 | 1 | 0 | 10.31 |
| −4 | 3 | 0.336 | 10.95 | −128 | 1 | 0 | 9.24 |
| −8 | 2 | 0.849 | 10.13 | −256 | 1 | 0 | 9.02 |
| −16 | 1 | 0 | 10.23 | −512 | 1 | 0 | 9.02 |

**Table 18**
The accuracy and speed of the MCL algorithm when applied to the Gordon data set with various parameters. The expansion parameter used in these runs was set to 2.

| $\delta$ | $r=0.5$ | | | $r=1$ | | | $r=4$ | | |
|---|---|---|---|---|---|---|---|---|---|
| | $K$ | $R$ | $T$ | $K$ | $R$ | $T$ | $K$ | $R$ | $T$ |
| 0.49 | 48 | 0.334 | 8.39 | 48 | 0.334 | 8.67 | 96 | 0.01 | 8.88 |
| 0.54 | 13 | 0.826 | 9.13 | 13 | 0.826 | 9.77 | 84 | 0.018 | 10.45 |
| 0.57 | 7 | -0.046 | 9.06 | 8 | 0.875 | 9.66 | 70 | 0.021 | 10.12 |
| 0.61 | 2 | -0.009 | 9.08 | 3 | 0.951 | 9.84 | 22 | 0.304 | 9.97 |

**Table 19**
The accuracy and speed of the normalized spectral clustering algorithm when applied to the Gordon data set with various parameters. In these runs, fully connected graphs were used.

| $\delta$ | $H=1$ | | | $H=5$ | | | $H=10$ | | |
|---|---|---|---|---|---|---|---|---|---|
| | $K$ | $R$ | $T$ | $K$ | $R$ | $T$ | $K$ | $R$ | $T$ |
| 6.3 | 5 | 0 | 43.69 | 5 | 0.75 | 40.45 | 5 | 0.75 | 40.19 |
| 11.61 | 5 | 0 | 42.59 | 5 | 0.99 | 47.6 | 5 | 0.99 | 42.48 |
| 16.15 | 5 | 0 | 53.23 | 5 | 1 | 41.34 | 5 | 0.99 | 41.41 |
| 30.38 | 5 | 0 | 30.45 | 5 | 1 | 30.07 | 5 | 0.98 | 30.44 |

**Table 20**
The accuracy and speed of the PP algorithm when applied to the multiple feature data set with the triangle kernel and various parameters.

| $\delta$ | $s=1$ | | | $s=100$ | | | $s=1000$ | | |
|---|---|---|---|---|---|---|---|---|---|
| | $K$ | $R$ | $T$ | $K$ | $R$ | $T$ | $K$ | $R$ | $T$ |
| 22.71 | 65 | 0.73 | 320.72 | 48 | 0 | 380.14 | 48 | 0 | 350.81 |
| 26.9 | 13 | 0.39 | 311.38 | 7 | 0 | 346.33 | 7 | 0 | 319.5 |
| 29.18 | 4 | 0 | 320.34 | 4 | 0 | 305.5 | 1 | 0 | 324.48 |
| 31.99 | 2 | 0 | 301.06 | 2 | 0 | 297.38 | 2 | 0 | 330.86 |

**Table 21**
The accuracy and speed of the MCL algorithm when applied to the multiple feature data set with various parameters. The expansion parameter used in these runs was set to 2.

| $\delta$ | $r = 0.5$ | | | $r = 1$ | | | $r = 4$ | | |
|---|---|---|---|---|---|---|---|---|---|
| | $K$ | $R$ | $T$ | $K$ | $R$ | $T$ | $K$ | $R$ | $T$ |
| 22.71 | 1 | 0 | 18.42 | 1 | 0 | 18.64 | 1 | 0 | 26.12 |
| 26.9 | 1 | 0 | 20.41 | 1 | 0 | 18.35 | 1 | 0 | 26.37 |
| 29.18 | 1 | 0 | 19.72 | 1 | 0 | 20.33 | 1 | 0 | 23.85 |
| 31.99 | 1 | 0 | 18.72 | 1 | 0 | 24.35 | 1 | 0 | 16.9 |

**Table 22**
The accuracy and speed of the normalized spectral clustering algorithm when applied to the multiple feature data set with various parameters. In these runs, fully connected graphs were used.

| $\delta$ | $H = 1$ | | | $H = 5$ | | | $H = 10$ | | |
|---|---|---|---|---|---|---|---|---|---|
| | $K$ | $R$ | $T$ | $K$ | $R$ | $T$ | $K$ | $R$ | $T$ |
| 22.71 | 10 | 0 | 292.12 | 10 | 0.32 | 283.97 | 10 | 0.37 | 287 |
| 26.9 | 10 | 0 | 291.92 | 10 | 0.32 | 285.29 | 10 | 0.37 | 281.46 |
| 29.18 | 10 | 0 | 294.12 | 10 | 0.32 | 278.57 | 10 | 0.37 | 287.11 |
| 31.99 | 10 | 0 | 281.56 | 10 | 0.32 | 287.81 | 10 | 0.37 | 284.77 |

slowest as this data set contains 2000 points. In the PP algorithm, each iteration involves matrix multiplication. In the normalized spectral clustering algorithm, matrix operation is conducted at the beginning of the algorithm. For this reason, it is expected that the PP algorithm is slower for large data sets.

We applied the PP algorithm and the MCL algorithm to a subset of the C-Cube test data set [5,6]. The subset contains the first 2000 records of the C-Cube test data set. Each record is described by 34 features extracted from a bitmap of a character. We used the Euclidean distance to measure the dissimilarity between two records. The accuracy and speed of the two clustering algorithms are shown in Tables 23 and 24. Comparing the Rand indices given Tables 23 and 24, we see that the PP algorithm is able to detect some cluster structures of this data set and that the MCL algorithm assigned all records into one cluster in all cases. Since the Rand indices of both algorithms are close to zero, it seems that the PP algorithm and MCL algorithm are not suitable for the C-Cube data set. This might be caused by the fact that the distance measure is not able to capture all the differences of the records. Subspace clustering algorithms (e.g., SAP [16], AFG-$k$-means [17], LEKM [14]) might be able to produce better clustering results for the C-Cube data set.

**Table 23**
The accuracy and speed of the PP algorithm when applied to the C-Cube data set with the triangle kernel and various parameters.

| $\delta$ | $s = 1$ | | | $s = 100$ | | | $s = 1000$ | | |
|---|---|---|---|---|---|---|---|---|---|
| | $K$ | $R$ | $T$ | $K$ | $R$ | $T$ | $K$ | $R$ | $T$ |
| 0.28 | 95 | 0.05 | 43.46 | 84 | 0 | 113.57 | 85 | 0.03 | 90.77 |
| 0.33 | 22 | 0.02 | 43.18 | 18 | 0 | 88.26 | 18 | 0 | 89.89 |
| 0.35 | 9 | 0.02 | 43.21 | 6 | 0 | 77.9 | 6 | 0 | 80.74 |
| 0.4 | 5 | 0 | 33.97 | 4 | 0 | 48.57 | 4 | 0 | 59.24 |

**Table 24**
The accuracy and speed of the MCL algorithm when applied to the C-Cube data set with various parameters. The expansion parameter used in these runs was set to 2.

| $\delta$ | $r = 0.5$ | | | $r = 1$ | | | $r = 4$ | | |
|---|---|---|---|---|---|---|---|---|---|
| | $K$ | $R$ | $T$ | $K$ | $R$ | $T$ | $K$ | $R$ | $T$ |
| 0.28 | 1 | 0 | 19.56 | 1 | 0 | 20.68 | 1 | 0 | 23.37 |
| 0.33 | 1 | 0 | 21.32 | 1 | 0 | 22.73 | 1 | 0 | 23.04 |
| 0.35 | 1 | 0 | 20.39 | 1 | 0 | 22.66 | 1 | 0 | 22.57 |
| 0.4 | 1 | 0 | 22.31 | 1 | 0 | 21.57 | 1 | 0 | 12.53 |

## 5. Concluding remarks

In this paper, we proposed a clustering algorithm, called the PP algorithm, by propagating probabilities between data points. The proposed algorithm uses two parameters: the bandwidth parameter $\delta$ and the parameter $s$. The bandwidth parameter is used to calculate the local densities of the data points in order to initialize the stochastic matrix. In general, a larger value of $\delta$ leads to less number of clusters. The parameter $s$ controls the shape of the clusters by limiting the number of attractors or exemplars a data point can choose. If a data set contains non-spherical clusters, a large value of $s$ leads to non-spherical clusters. If a data set contains only spherical clusters, then the clustering result is relatively independent of $s$.

Similar to the AP algorithm and the MCL algorithm, the PP algorithm is a message-passing algorithm. One major difference of the PP algorithm and the AP algorithm is that the message-passing rules of the PP algorithm are simpler than those of the AP algorithm. The PP algorithm is also different from the MCL algorithm in two ways: first, the PP algorithm does not use the inflation parameter to control the number of clusters but the MCL algorithm does; second, the stochastic matrix initialization of the PP algorithm is different from that of the MCL algorithm.

In future, we would like to extend this work in the following directions. First, we would like to explore other methods to initialize the probabilities in the stochastic matrix. In the proposed algorithm, we used local densities to initialize the probabilities. Second, we would like to investigate ways to speed up the proposed algorithm as the computation complexity is $O(n^3)$. Third, we would like to extend the proposed algorithm to identify subspace clusters [14,16,17].

## References

[1] C.C. Aggarwal, C.K. Reddy (Eds.), Data Clustering: Algorithms and Applications, CRC Press, Boca Raton, FL, USA, 2013.
[2] A.A. Alizadeh, M.B. Eisen, R.E. Davis, C. Ma, I.S. Lossos, A. Rosenwald, J.C. Boldrick, H. Sabet, T. Tran, X. Yu, J.I. Powell, L. Yang, G.E.M. Troy Moore, J. James Hudson, L. Lu, D.B. Lewis, R. Tibshirani, G. Sherlock, W.C. Chan, T.C. Greiner, O. James, D.D.W. Armitage, R. Warnke, R. Levy, W. Wilson, M.R. Grever, J.C. Byrd, D. Botstein, P.O. Brown, L.M. Staudt, Distinct types of diffuse large b-cell lymphoma identified by gene expression profiling, Nature 403 (2000) 503L 511.
[3] A. Banerjee, S. Merugu, I.S. Dhillon, J. Ghosh, Clustering with Bregman divergences, J. Mach. Learn. Res. 6 (2005) 1705–1749.
[4] L. Bregman, The relaxation method of finding the common point of convex sets and its application to the solution of problems in convex programming, USSR Comput. Math. Math. Phys. 7 (3) (1967) 200–217.
[5] F. Camastra, M. Spinetti, A. Vinciarelli, Cursive character challenge: a new database for machine learning and pattern recognition, in: Proceedings of International Conference on Pattern Recognition (ICPR), 2006, pp. 1–9.
[6] F. Camastra, A. Vinciarelli, Cursive character recognition by learning vector quantization, Pattern Recognit. Lett. 22 (6-7) (2001) 625–629.
[7] M. de Souto, I. Costa, D. de Araujo, T. Ludermir, A. Schliep, Clustering cancer gene expression data: a comparative study, BMC Bioinform. 9 (1) (2008) 1–14.
[8] I.S. Dhillon, Y. Guan, B. Kulis, Kernel $k$-means: spectral clustering and normalized cuts., in: Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '04, ACM, New York, NY, USA, 2004, pp. 551–556.
[9] M. Filippone, F. Camastra, F. Masulli, S. Rovetta, A survey of kernel and spectral methods for clustering, Pattern Recognit. 41 (1) (2008) 176–190.
[10] A. Frank, A. Asuncion, UCI Machine Learning Repository, 2010, Available at: http://archive.ics.uci.edu/ml.
[11] B.J. Frey, D. Dueck, Clustering by passing messages between data points? Science 315 (5814) (2007) 972–976.
[12] G. Gan, Data Clustering in C++: An Object-Oriented Approach. Data Mining and Knowledge Discovery Series, Chapman & Hall/CRC Press, Boca Raton, FL, USA, 2011.
[13] G. Gan, Application of data clustering and machine learning in variable annuity valuation? Insur.: Math. Econ. 53 (3) (2013) 795–801.
[14] G. Gan, K. Chen, A soft subspace clustering algorithm with log-transformed distances, Big Data Inf. Anal. 1 (1) (2016) 93–109.
[15] G. Gan, S. Lin, Valuation of large variable annuity portfolios under nested simulation: a functional data approach, Insur.: Math. Econ. 62 (2015) 138L 150.
[16] G. Gan, M.K.-P. Ng, Subspace clustering using affinity propagation? Pattern Recognit. 48 (4) (2015) 1451–1460.

[17] G. Gan, M.K.-P. Ng, Subspace clustering with automatic feature grouping? Pattern Recognit. 48 (11) (2015) 3703–3713.

[18] G.J. Gordon, R.V. Jensen, L.-L. Hsiao, S.R. Gullans, J.E. Blumenstock, S. Ramaswamy, W.G. Richards, D.J. Sugarbaker, R. Bueno, Translation of microarray data into clinically relevant cancer diagnostic tests using gene expression ratios in lung cancer and mesothelioma? Cancer Res. 62 (17) (2002) 4963–4967.

[19] A. Jain, Data clustering: 50 years beyond *k*-means? Pattern Recognit. Lett. 31 (8) (2010) 651–666.

[20] A. Jain, R. Duin, J. Mao, Statistical pattern recognition: a review? IEEE Trans. Pattern Anal. Mach. Intell. 22 (1) (2000) 4–37.

[21] A. Jain, M. Murty, P. Flynn, Data clustering: a review? ACM Comput. Surv. 31 (3) (1999) 264–323.

[22] R. Liu, A. Anand, D.K. Dey, B. Javidi, Entropy-based clustering of embryonic stem cells using digital holographic microscopy? J. Opt. Soc. Am. A 31 (4) (2014) 677–684.

[23] U. Luxburg, A tutorial on spectral clustering? Stat. Comput. 17 (4) (2007) 395–416.

[24] J. Macqueen, Some methods for classification and analysis of multivariate observations, in: L. LeCam, J. Neyman (Eds.), Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability, Vol. 1, University of California Press, Berkely, CA, USA, 1967, pp. 281–297.

[25] A.Y. Ng, M.I. Jordan, Y. Weiss, On spectral clustering: Analysis and an algorithm, in: T. Dietterich, S. Becker, Z. Ghahramani (Eds.), Advances in Neural Information Processing Systems 14, The MIT Press, 2002, pp. 849–856.

[26] J. Peña, J. Lozano, P. Larrañaga, An empirical comparison of four initialization methods for the *k*-means algorithm, Pattern Recognit. Lett. 20 (10) (1999) 1027–1040.

[27] J. Shi, J. Malik, Normalized cuts and image segmentation? IEEE Trans. Pattern Anal. Mach. Intell. 22 (8) (2000) 888–905.

[28] B.W. Silverman, Density Estimation for Statistics and Data Analysis, CRC Press, Boca Raton, FL, 1986.

[29] S. van Dongen, Graph Clustering by Flow Simulation, (PhD thesis), University of Utrecht, 2000.

[30] S. van Dongen, Graph clustering via a discrete uncoupling process? SIAM J. Matrix Anal. Appl. 30 (1) (2008) 121–141.

[31] A.D. Well, J.L. Myers, Research Design & Statistical Analysis, 2nd ed., Psychology Press, Mahwah, NJ, 2003.

[32] R. Xu, D. Wunsch, Clustering, Wiley-IEEE Press, Hoboken, NJ, 2008.

[33] N. Yilmaz, O. Inan, M. Uzer, A new data preparation method based on clustering algorithms for diagnosis systems of heart and diabetes diseases? J. Med. Syst. 38 (5) (2014) 1–12.