Pattern Recognition 86 (2019) 265–280

Contents lists available at ScienceDirect

Pattern Recognition

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

# Clustering as physically inspired energy minimization

Huiguang Yang [a,b,*], Narendra Ahuja [a]

[a] Electrical and Computer Engineering, University of Illinois, 1520 Beckman Institute, 405 N.Mathews, Urbana, IL 61801, USA
[b] Samsung Research Institute China, Xi'an (SRCX), No. 156, 8th Tiangu Rd., Property B1, Xi'an, Shaanxi, 710077, China

ARTICLE INFO

*Article history:*
Received 11 April 2017
Revised 1 August 2018
Accepted 10 September 2018
Available online 20 September 2018

*Keywords:*
Unsupervised/hierarchical clustering
Energy minimization
Statistical physics
Integer programming
Unary/data energy
Ising model
Local density
Connected component analysis
Image segmentation
Normalized-cut

ABSTRACT

We formulate the task of density based clustering as energy minimization, using both binary/pairwise energy term and unary/data energy term (the latter was largely ignored in previous clustering methods). Binary energy is defined in terms of inhomogeneity in local point density. While most previous methods use binary/pairwise energy only, the unary/data energy can represent the natural tendency of a given point belonging to a given cluster, which is also crucial for the clustering. Since our energy is expressed as the sum of a unary (data) term and a binary (pairwise or smoothness) term, we can thus make a perfect analogy with the energy model used in vision and borrow everything (such as the optimization algorithms) from vision field to clustering under this mapping. This correspondence provides an entirely new view point in handling the clustering problem, and in fact many mature methods and algorithms are already provided in the vision field and can be adopted by the clustering field readily. During our energy optimization, a sequence of energy minima are found to recursively partition the points, and thus find a hierarchical embedding of clusters that are increasingly homogeneous in density. Disjoint clusters with the same density are identified separately. Our clustering method is totally unsupervised (which is superior to most existing methods, as those listed below). It does not need any user input parameters (say number of segments, any bandwidth parameter, cutoff distance/scale, etc.), except one can specify the *homogeneity criterion* — the degree of acceptable fluctuation in density within a cluster (which is target-related), or let it be specified automatically in a hierarchical way. We conduct experiments on both synthetic datasets and real-image tasks. Experimental results on synthetic datasets show that our method is able to handle clusters of different shapes, sizes and densities. We present the performance of our approach using the commonly used energy optimization algorithms from vision such as ICM, LBP, Graph-cut, Mean field theory algorithm, as well as the integer programming algorithm. We also contrast our performance with several other commonly used clustering algorithms, such as *k*-means, fussy *c*-means, DBSCAN, as well as a recent state-of-the-art clustering method as reported in [40]. Our experiments on real-image tasks further validate the performance of the method. In addition, we show that the family of commonly used spectral, graph clustering algorithms (such as Normalized-cut) uses only the binary energy term while ignoring the unary energy term; therefore, their energy model is incomplete compared with ours.



## 1. Introduction

This paper presents an approach to detection of clusters characterized by homogeneity of density inside and a discontinuity of density across border. We define an energy function that captures density variation and minimize it to find the clusters of the data points. This is in contrast with many methods summarized in literature [1–4], as well as the more recent ones in [40–49]. The recent literature [1] classifies clustering algorithms into traditional and modern ones. They divide the traditional algorithms into the following 9 classes, based on: (1) partition (2) hierarchy (3) fuzzy theory (4) distribution (5) density (6) graph theory (7) grid (8) fractal theory (9) model. They divide the modern clustering methods into the following 10 classes, based on: (1) kernel (2) ensemble (3) swarm intelligence (4) quantum theory (5) spectral graph theory (6) affinity propagation (7) density and distance (8) spatial data (9) data stream (10) large-scale data. For example, the popular *k*-means method [20] is based on partition, mean-shift method [21] and DBCSAN [25] are based on density, and the Normalized-cut method [22] is based on spectral graph theory.

Our approach applies ideas from the analysis of particle interactions done in statistical physics to associate an energy func-

* Corresponding author.
*E-mail addresses:* hyang30@illinois.edu, hg1.yang@samsung.com (H. Yang), n-ahuja@illinois.edu (N. Ahuja).

https://doi.org/10.1016/j.patcog.2018.09.008
0031-3203/


**Fig. 1.** An example to illustrate that two distant points may belong to the same cluster while two closer points may not.

tion with the network of data points, and perform graphical model energy optimization using Ising model [23] from physics. Desired clusters are then obtained corresponding to energy minima. Energy is defined as consisting of a unary/data energy term and another, binary/smoothness term capturing interactions among data points. Previous work [5–10] has also considered using concepts from statistical physics for the clustering problem, but they did so only partially. For example, Rose et al. [5], uses energy but defines it heuristically (e.g., based on square distance between points), instead of following the Ising model; in [6], the energy function does not contain the unary term, which led the authors to conclude that the energy function is symmetric in the global permutation of all labels. However, as we will discuss later, every data point has its natural tendency to belong to a certain cluster; for example, a data point located in a high density area is more likely to belong to the cluster with higher density and vice versa. With the inclusion of the unary energy term, we can quantify this natural tendency automatically.

A key issue our method accounts for when applying the energy optimization scheme to clustering problem is that we do not know the number of clusters in advance. In addition, we avoid defining clustering affinity among the data points based on their mutual distances, since being closer does not guarantee more likely belonging to the same cluster (Fig. 1).

### 1.1. The contributions

The contributions of this work as well as the essential difference with the previous "energy based" clustering schemes are as follows:

1. We more completely map the energy model of statistical physics onto clustering problem; especially, we emphasize the importance of the unary energy term in clustering. Therefore, our method can be totally unsupervised; which is not true for most previous methods where some key input parameters are needed, such as the number of segments, the bandwidth parameters, cutoff distance/scale, etc.
   In our work, we do not just introduce or emphasize the unary term in clustering, but more importantly we put the unary and binary energy terms in the principled way defined by statistical physics, not in some heuristic way. Therefore our entire scheme is systematic and principled, which also corresponds to the energy formulation used in computer vision immediately, and hence opens the doors for borrowing the mature concepts/algorithms/solutions in computer vision for the clustering problem as stated below.
2. Since our energy model consists of a unary (data) term and a binary (pairwise or smoothness) term, we can thus make a perfect analogy with the energy model used in vision and borrow the concepts, ideas, and schemes (such as the optimization algorithms) from vision field to clustering under this mapping. This correspondence was largely ignored in the clustering field and hence this mapping provides an entirely new view point for handling the clustering problem.
3. We propose a data point local density estimation method which can account for the datasets with arbitrary shapes and topologies (such as with holes inside).
4. We point out that the spectral clustering methods (such as Normalized-cut [22]) use only the binary/pairwise energy term while leaving out the unary/data energy term; therefore, their energy model is incomplete compared with ours.

### 1.2. The organization

The organization of this paper is as follows. In Section 2, we present our method, including determination of the point local density and its homogeneity, and the energy function with its unary and binary terms. In Section 3, we show that the energy model considered in the graph spectral clustering method (such as Normalized-cut) is incomplete compared with ours. Section 4 presents the experimental results we obtain on both synthetic datasets and real-image tasks, as well as the comparison with other related methods. Finally we conclude in Section 5.

## 2. Energy optimization based clustering

We draw an analogy with the *image segmentation* problem to formulate the energy based objective function. In image segmentation, the property of each node (image pixel or superpixel) that distinguishes it from other nodes (pixels) is the pixel intensity, and the adjacent pixels with similar intensities are more likely to belong to the same segment of the image. Analogously, in clustering problem, the property of each node (data point) that distinguishes it from other nodes is the point local density, and the adjacent data points with similar local densities are more likely to belong to the same cluster. The energy formulation for image segmentation may then be directly mapped onto the clustering problem.

Recall that in the image segmentation case, a graphical model [24] is built treating each image pixel (or superpixel) as a node $v_i$ in the graph. We aim to assign each node $v_i$ with a segment label $x_i$. The energy objective (1) defined on this graphical model typically consists of a *unary (data) term* and a *binary (smoothness) term*, representing the nodeâs local information (compatibility with labeling) and the nodes interaction information (compatibility of labels at a pair of neighboring nodes). $N$ denotes the neighborhood set in (1).

$$E = \sum_i E_i^{data}(x_i) + \sum_{(i,j)\in N} E_{ij}^{smoothness}(x_i, x_j) \tag{1}$$

Analogously, for the clustering problem we define a graphical model on the point set, where each node (data point) is connected with its (suitably defined) neighbors in the graph (see Section 2.1). The energy objective can be formulated in the same manner as in the image segmentation case, using data and smoothness terms.

We will first consider the case of binary labeling. This is because doing so will simplify analysis without loss of generality, since the multi-label problem can be solved via a sequence of binary labeling problems. For example, for multi-region image segmentation, we can recursively split the image (and some of its sub-images) into two groups of segments, when necessary; similarly, for clustering problem, we can recursively split the point set (and some of its sub-sets) into two parts, if needed. In the binary

labeling case, let us denote the labels as $x_i \in \{-1, 1\}$ for node $v_i$. Then, based on the principles of statistical physics, the energy can be written in the following form.

$$E = \sum_i \theta_i x_i - \sum_{(i,j)\in N} w_{ij} x_i x_j \tag{2}$$

where $\theta_i$ is the unary parameter and $w_{ij}$ is the binary edge weight.

This formulation is the well-known Ising model [23] in statistical physics, which studies a system of large number of interacting particles that is conceptually similar with our problem. Also, it is straightforward to show that the commonly used Potts model [28]:

$$\begin{aligned}\sum_{(i,j)\in N} E_{ij}^{smoothness}(x_i, x_j) &= Potts(x)\\ &= \sum_{(i,j)\in N} I(x_i \neq x_j) w_{ij}\\ &= \frac{1}{2}(-\sum_{(i,j)\in N} w_{ij} x_i x_j + \sum_{(i,j)\in N} w_{ij})\end{aligned} \tag{3}$$

for the smoothness term in computer vision problem can be converted into the Ising model form (2) (possibly up to a scaling and a constant), where $I(\cdot)$ is the indicator function with value 1 when its statement true and value 0 when its statement false. In addition, the unary parameter $\theta_i$ in the data term in (2) can be expressed as follows (consider $E_i^{data}(x_i) = \theta_i x_i$ ).

$$\theta_i = \frac{1}{2}(E_i^{data}(x_i = 1) - E_i^{data}(x_i = -1)) \tag{4}$$

This is the unary energy difference for labeling the node $v_i$ with $+1$ versus $-1$.

### 2.1. Determination of local density

The local density around each point in the data set is the key factor in forming the clusters [25,26]. Therefore, determining the point local density becomes the key issue in the clustering process. The first question we encounter is what is a good notion of *neighborhood*. There have been various definitions of neighborhood in the literature, including the fixed scale/radius around the given point, the k-nearest neighborhood (k-NN), the minimal spanning tree based neighborhood, etc. Among all those definitions, one particular natural and meaningful notion is the *Delaunay neighborhood*, obtained using the Delaunay triangulation [29] of the point set. Neighboring points are the ones connected by the triangulation, and the number of these neighbors can vary from point to point. As pointed out by Ahuja [27], a crucial difference between the Delaunay and most traditional definitions of neighborhood is that the former makes use of the Euclidean plane and reflects local structure, while most traditional definitions above determine the neighborhood directly in terms of inter-point distances. It is also pointed out in [27] that, compared with the fixed scale neighborhood, the advantages of the Delaunay neighborhood would include being sensitive to variations in the local densities and other properties of local point distributions; and compared with k-nearest neighborhood, the advantages would include the number of neighbors can vary from point to point, as well as neighbors being symmetric (k-NN neighbors may not be symmetric).

It should be noted that the Delaunay neighbors are not necessarily the nearest neighbors, e.g., a point at the boundary of a cluster can connect with far away points, in which case the resulting local density computed based on the Delaunay neighbors can be very inaccurate or even meaningless. Hence we do not directly rely on the Delaunay neighborhood determined at a given point to compute the local density; instead, we still consider the k-nearest neighborhood but with a varying $k$, which is naturally determined by the number of Delaunay neighbors found at each point (or some value proportional to it). In the region where the points are distributed uniformly, the Delaunay neighbors would typically coincide with the k-nearest neighbors. In such manner the proper choice of $k$ at each point can be determined automatically.

Given the proper value of $k$ and k-NN around each point, we compute the distance between each neighbor in the k-NN region to the central point, and take the *median* distance among them as the *characteristic distance* ($d_c$) for the central point (see (5)). We assume the local density is inversely proportional to this characteristic distance (specifically, to the *nth*-order of the characteristic distance in $n$-dimensional space; see Fig. 2 for examples):

$$d_c(v_i) = median(d(v_i, v_j) : v_j \in kNN(v_i)) \tag{5}$$

$$\rho(v_i) \propto \frac{1}{d_c(v_i)}, \tag{6}$$

where $d(v_i, v_j)$ is the spatial distance between the point $v_i$ and $v_j$, $kNN(v_i)$ is the k-nearest neighborhood of point $v_i$, $d_c(v_i)$ is the characteristic distance of point $v_i$, and $\rho(v_i)$ is the local density at point $v_i$. Given the local density $\rho(v_i)$ at each point, we can then treat this feature in the same way as the pixel intensity in the image segmentation case, and analogously compute the data (unary) and smoothness (binary) energy terms.

### 2.2. Data (unary) term

Given the local density $\rho(v_i)$ at every point $v_i$ in the data set, we can simply find the maximal local density $\rho_{\max}$ and minimal local density $\rho_{\min}$ . In the binary labeling case, we want to cluster all the points into two clusters; one is relatively dense and the other relatively sparse, and each one of them can be similarly further divided into two, if needed. Hence, for any given point $v_i$, the closer its local density $\rho(v_i)$ to $\rho_{\max}$, the more likely (i.e. lower data energy) it belongs to the dense cluster, and vice versa. If we assign the dense cluster the label $+1$ and sparse cluster the label $-1$, we compute the data (unary) energy for each point $v_i$ as follows:

$$\begin{aligned}E_i^{data}(x_i = 1) &= \frac{\rho_{max} - \rho(v_i)}{\rho_{max} - \rho_{min}} - 1/2\\ E_i^{data}(x_i = -1) &= \frac{\rho(v_i) - \rho_{min}}{\rho_{max} - \rho_{min}} - 1/2.\end{aligned} \tag{7}$$

Therefore, the unary parameter $\theta_i$ can be obtained as in (4) (we add $-1/2$ in (7) to let $E_i^{data}(x_i = 1) = -E_i^{data}(x_i = -1)$ and it has no impact on computing $\theta_i$). In fact, a large positive $\theta_i$ means the unary energy for labeling the node $v_i$ with label $+1$ is much higher than that with label $-1$, or the likelihood of node $v_i$ belonging to the dense cluster (label $+1$) is much lower than that of belonging to the sparse cluster (label $-1$), so the node tends to be labeled with $x_i = -1$ . Therefore, with this labeling, the unary energy $E_i^{data}(x_i) = \theta_i x_i$ is negative (and large positive for $x_i = 1$ ), as would be desired.

### 2.3. Smoothness (binary) term

In image segmentation, the edge weight $w_{ij}$ between the node $v_i$ and $v_j$ (representing image pixels or superpixels) is computed in terms of the intensity contrast between the two nodes:

$$w_{ij} = exp(-\frac{(I(v_i) - I(v_j))^2}{2\sigma^2}) \tag{8}$$

That is, larger the intensity difference, weaker the edge connection. In the exactly same way, we can compute the edge weight

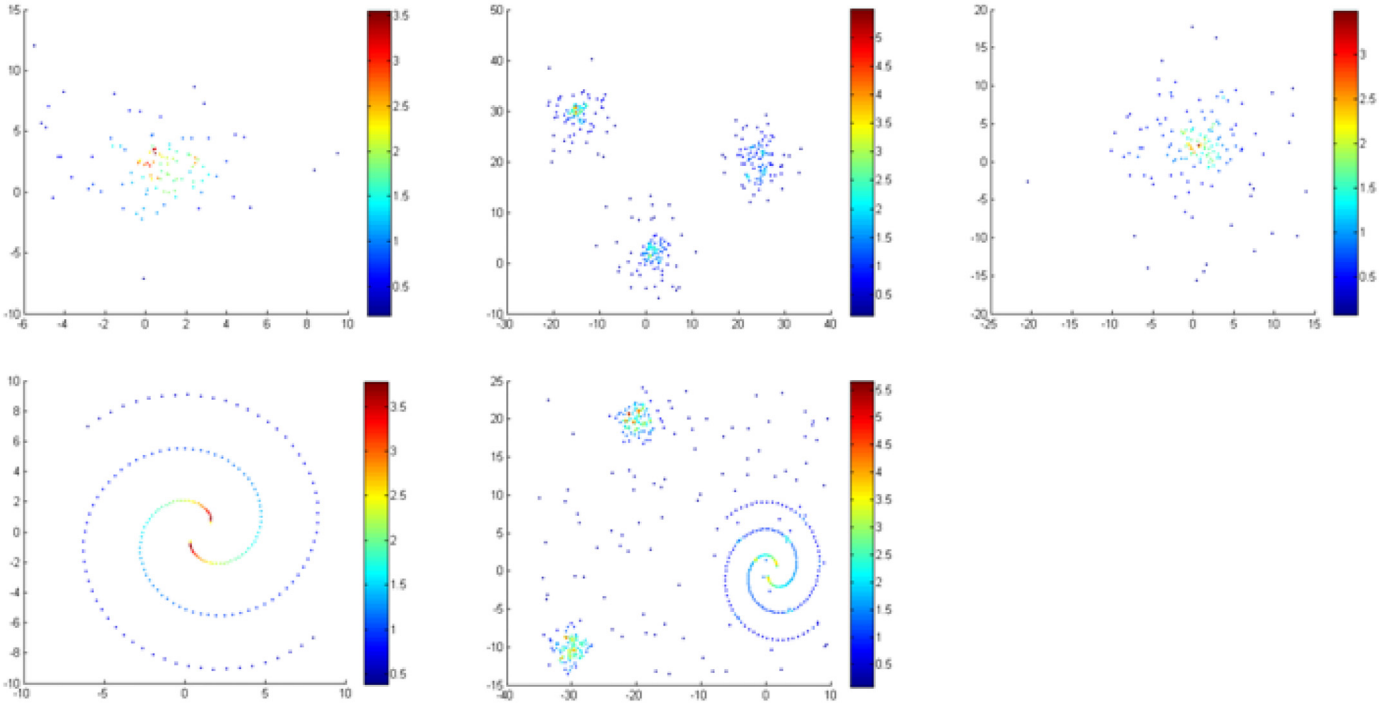

**Fig. 2.** Computed point local density for various datasets.

between two neighboring nodes in the clustering case, in terms of the local density contrast between the nodes:

$$w_{ij} = exp(-\frac{(\rho(v_i) - \rho(v_j))^2}{2\sigma_\rho^2}) \tag{9}$$

Again, the larger the local density difference, the weaker the edge connection between the two nodes.

Given the data and smoothness energy terms, the overall energy objective of the clustering problem can be written by directly following the Ising model formulation in statistical physics (ref. (2)):

$$E = \sum_i \theta_i x_i - \sum_{(i,j)\in N} w_{ij} x_i x_j. \tag{10}$$

The formulation is based on the unary/binary parameters $\theta_i$ and $w_{ij}$ as defined above, which in turn are functions of the point local density $\rho(v_i)$.

### 2.4. Connected component analysis

The clusters obtained so far cannot separate spatially isolated clusters associated with the similar density value. To identify these spatially separate clusters, we perform connected component analysis [30–32] on each obtained cluster and separate any isolated subgroups of points therein. A key parameter in running the connected component analysis is the distance between two points considered as "close enough" or "connected". For this, we use the characteristic distance $d_c^{cluster}$ of the current cluster; specifically, we choose the largest characteristic distance among all the nodes in the current cluster:

$$d_c^{cluster} = \max_i\{d_c(v_i), v_i \in \text{current} \quad \text{cluster}\} \tag{11}$$

### 2.5. Density homogeneity measure of cluster

One final question is how to measure the local density homogeneity of a cluster, some of which are presented in [33]. In our case, we use the ratio of local density standard deviation $std(\rho)$ and average density $mean(\rho)$ for a given cluster as the homogeneous criterion $h_{cr}$:

$$h_{cr} = std(\rho)/mean(\rho) \tag{12}$$

.

The lower the value of $h_{cr}$, the more homogeneous the cluster. Since this homogeneity measure is point-based, it can handle the cluster with any shape and topology. In our experiments, e.g., we work with clusters of spiral-shape and containing holes.

### 2.6. The overall methodology

To summarize, the main steps in our hierarchical energy optimization based clustering are as follows (also see Table 1).

1. We first compute the local density at each point. (a) Using the number of Delaunay neighbors to determine the value of k (in k-NN algorithm) at each data point; (b) Using Eqs. (5) and (6) to compute the characteristic distance and the local density at each data point, respectively.
2. Based on the local density values, we use Eq. (12) to measure the density homogeneity of the data set or the given cluster.
3. We apply the energy based clustering algorithm, if the local density is not sufficiently homogeneous, to divide the point set into two clusters, one dense and one relatively sparse. (a) Based on the local density value at each data point, obtaining the maximal local density and the minimal local density for the given data set; (b) Using Eqs. (4) and (7) to compute the unary energy term for the objective energy function; (c) Using Eq. (9) to compute the edge weights and hence the binary energy term for the objective energy function; (d) Using Eq. (10) to compute the overall energy function for optimization.
4. We apply the connected component analysis on the current point set, if the local density across the current point set is sufficiently homogenous, to separate it into all disconnected clusters if more than one. (a) Using Eq. (11) to obtain the key parameter for performing connected component analysis.

**Table 1**
Hierarchical energy optimization based clustering algorithm.

| Algorithm Hierarchical energy optimization based clustering |
|---|
| **Input** Data set X (no more tuning parameters needed).<br>1: Determine the k value at each data point for the k-NN algorithm, based on the<br>number of Delaunay neighbors (or for simplicity set k=5 or some other constant, experimentally working well in many cases).<br>2: Compute the characteristic distance and hence the local density at each data point.<br>3: Evaluate the density homogeneity of the data set, based on the local density values<br>therein.<br>4: **Loop until** the clusters at hand are all sufficiently homogeneous (and all checked by (or obtained from) connected component analysis).<br>5: **If** the local density is not sufficiently homogeneous across the current data set<br>Apply the energy based clustering algorithm to bi-partition the current data set or cluster.<br>6: **Else**<br>Apply the connected component analysis on the current data set or cluster, to<br>divide it into all disconnected clusters if more than one.<br>7: **End if**<br>8: Evaluate the density homogeneity of each obtained cluster, based on their local density values therein.<br>9: **End loop**<br>**Output Clustering C of X.** |

5. We apply the algorithm recursively for each of the obtained two (or more) clusters.
6. These recursions continue until the clusters at hand are all sufficiently homogeneous (and all checked by (or obtained from) connected component analysis).

Our method is essentially parameter-free. The only parameters in our algorithm formulation (Eqs. (4)–(7) and Eqs. (9)–(12)) that need to be determined are the $k$ value in the k-NN algorithm in Eq. (5) and the density standard deviation $\sigma_\rho$ in Eq. (9). The $k$ value can be determined via the number of Delaunay neighbors, or in practice we notice that in many problems it can be simply set as a constant value (such as $k = 5$ ). As for the density standard deviation $\sigma_\rho$ in Eq. (9), we can simply set it as $\sigma_\rho = 0.1(\rho_{\max} - \rho_{\min})$ and it works well in most cases.

## 3. The connections with Graph spectral based clustering approach

In this section, we relate our algorithm with the graph spectral method [34] because it shares some important characteristics with our algorithm. They are both based on the graphical model structure connecting all the points, and both also consider interaction among them via edge weight or adjacency matrix. We show that the graph spectral method (such as Normalized-cut [22]) uses only the binary or smoothness term in the energy model, and ignores the unary or data term; therefore, their energy model is incomplete compared with ours.

Given a graphical model representation of the problem, the graph spectral method finds the solution through eigen-analysis of graph-associated matrices such as the adjacency or Laplacian matrix [36]. Consider a graphical network $G = (V, E)$, where $V$ and $E$ are the node set and edge set, respectively, with a total of $n$ nodes. We denote the edge weight matrix of the graph by $W$, where $w_{ii} = 0$ . Then the degree matrix $D$ of the graph can be defined accordingly, with node degree $d$ satisfying the equation:

$$d_i = \sum_{(i,j)\in N} w_{ij} \tag{13}$$

at the diagonal.

The graph Laplacian matrix $L$ is then defined as in (14)

$$L = D - W, \tag{14}$$

which is a positive semidefinite matrix [34,35].

Let vector $x \in \{-1, 1\}^n$, then the quadratic form of the graph Laplacian is

$$x^T L x = x^T (D - W)x = -\sum_{i,j,i\neq j} w_{ij}x_i x_j + \sum_i d_i \tag{15}$$

If we ignore the constant term $\Sigma_i d_i$ and note that $w_{ij} = 0$ for $(i, j) \notin N$ , then the RHS above is the familiar smoothness energy term in Ising model (2). In fact, the formulation in (2) can be rewritten as

$$E = \sum_i \theta_i x_i - \sum_{i,j,i\neq j} w_{ij}x_i x_j = \theta^T x + x^T L x - \sum_i d_i, \tag{16}$$

where $L$ is the graph Laplacian matrix for the problem. Therefore, the physics based energy minimization problem (2) becomes

$$\text{argmin}_{x, x_i\in\{-1,1\}} \theta^T x + x^T L x \tag{17}$$

In contrast to (17), the graph spectral algorithms minimize only the (binary or smoothness) term $x^T L x$ or its variants, and do not consider the (unary or data) term $\theta^T x$. Below we will illustrate this point with a concrete example of the Normalized-cut method.

A graph $G = (V, E)$ can be partitioned into two disjoint sets $A$ and $B$ by simply removing the edges between the two parts. The *cut* and *association* between the node set $A$ and $B$ are defined as

$$cut(A, B) = \sum_{i\in A, j\in B} w_{ij}, \quad assoc(A, V) = \sum_{i\in A, j\in V} w_{ij} \tag{18}$$

If we denote the node labels as $x_i \in \{1, -1\}$ for bi-partitioning the graph into the set $A$ and $B$, and the node degree as $d_i$, the Normalized-cut formulation can be written as

$$Ncut(x) = \frac{\sum_{x_i=1, x_j=-1} -w_{ij}x_i x_j}{\sum_{x_i=1} d_i} + \frac{\sum_{x_i=-1, x_j=1} -w_{ij}x_i x_j}{\sum_{x_i=-1} d_i} \tag{19}$$

It is clear that the numerator in (19) is closely related with the binary (smoothness) energy term in (2). Specifically, it only sums up the binary terms where the labels for the neighboring nodes are different, which makes it similar to the Potts model [28]:

$$Potts(x) = \sum_{x_i=1, x_j=-1} -w_{ij}x_i x_j + \sum_{x_i=-1, x_j=1} -w_{ij}x_i x_j \tag{20}$$

Since $cut(A, B) = cut(B, A)$, the numerators in (19) are equal to the half of the Potts model, making the Normalized-cut formula closely related with the Potts model as follows:

$$\begin{aligned} Ncut(x) &= \frac{\sum_{x_i=1, x_j=-1} -w_{ij}x_i x_j}{\sum_{x_i=1} d_i} + \frac{\sum_{x_i=-1, x_j=1} -w_{ij}x_i x_j}{\sum_{x_i=-1} d_i} \\ &= \frac{1}{2}\left(\frac{1}{\alpha(x)} + \frac{1}{\beta(x)}\right) Potts(x), \end{aligned} \tag{21}$$

where $\alpha(x) = \sum_{x_i=1} d_i$ and $\beta(x) = \sum_{x_i=-1} d_i$ are functions of the labeling, but they only consider the edge connections among the nodes (namely, the node degree $d_i$), while having nothing to do with the local unary property at the node (such as the local density at the node for the clustering case, ref. (7) and (4)), which

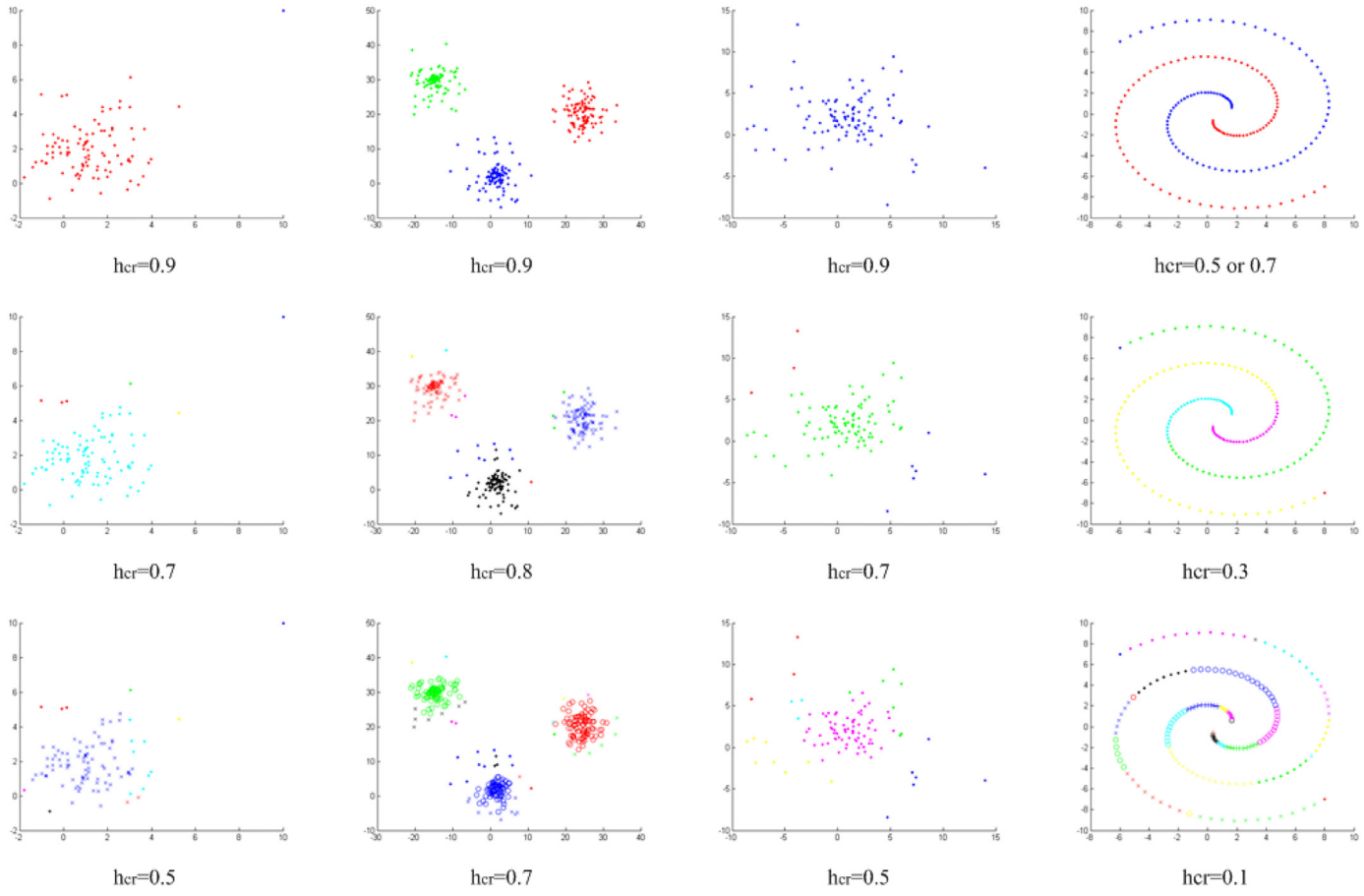


**Fig. 3.** Clusters obtained by our method for various types of point sets (each column) with different values of the homogeneity criterion $h_{cr}$.

represents the natural tendency of a node belonging to a certain label class. Therefore the formulation in (21) is unrelated with the unary/data energy term as we described in our model.

Hence, the Normalized-cut formulation includes only the binary or smoothness term in the energy function, while ignoring the unary or data term. The performance of Normalized-cut method will be limited due to the lack of unary term. Section 4.2 provides the experimental results in support of this observation.

## 4. Experiments

### 4.1. Experiments on synthetic dataset

We first evaluate our method on some synthetic datasets. In our experiment, we evaluate the performance of our clustering method and compare it with other commonly used and best-performing methods (Sections 4.1.1 and 4.1.2). Meanwhile, we compare the performance of various specific energy optimization algorithms within our scheme (Sections 4.1.3). It should be noted that most previous methods require some heuristic user input for the clustering (such as cluster number, distance scale or minimum number of points to form the cluster etc., as needed in $k$-means and DBSCAN methods), our method does not need any of those inputs; hence, our method is purely unsupervised.

#### 4.1.1. The experiments on various types of datasets

In this subsection, we test our clustering algorithm on various types of datasets, containing clusters with different shapes and topologies. The effect of the homogeneity criterion $h_{cr}$ on the clustering is also tested. We compare our algorithm with the most widely used and best-performing methods in the field (such as $k$-means and DBSCAN), along with a recent state-of-the-art method [40], and also illustrate our performance on some very challenging datasets that recent literature [3] notes no method can handle it well. We use the ICM algorithm [11] for energy optimization in this experiment.

From the experimental results shown in Fig. 3, we can see that when the value of homogeneity criterion $h_{cr}$ decreases, the point set will be divided into more and more clusters, with increasingly homogeneous densities. For example, for the spiral shape point set in Fig. 3, we note that the point density along each spiral branch gradually decreases from the center to the outside region. When the homogeneity criterion large ($h_{cr} = 0.5$ or 0.7), each spiral branch is separated into a distinct cluster; when the homogeneity criterion becomes smaller ($h_{cr} = 0.1$ or 0.3), the point density fluctuation along each spiral branch is no longer tolerated, and each branch will thus be further split into multiple clusters.

In Fig. 4, we have produced a composite dataset containing various point cloud types appeared above, with a spiral, two dense-core point clouds, and some sparse scatter points all over the background. This type of dataset is inspired by Fig. 2 in [3], demonstrating the significant variation of the cluster type, shape, density, size etc. The author in [3] claimed that none of the available clustering methods can handle this type of dataset well and detect all the clusters. Yet we see our clustering method offers a reasonable hierarchical clustering of the dataset. Our results show that when the homogeneity criterion is large $h_{cr} = 1.1$ (i.e. large local density fluctuation is allowed), all the points are grouped into the same cluster. When $h_{cr} = 1.0$, the three major intuitive clusters, the spiral and two dense-core point clouds, are clearly separated;

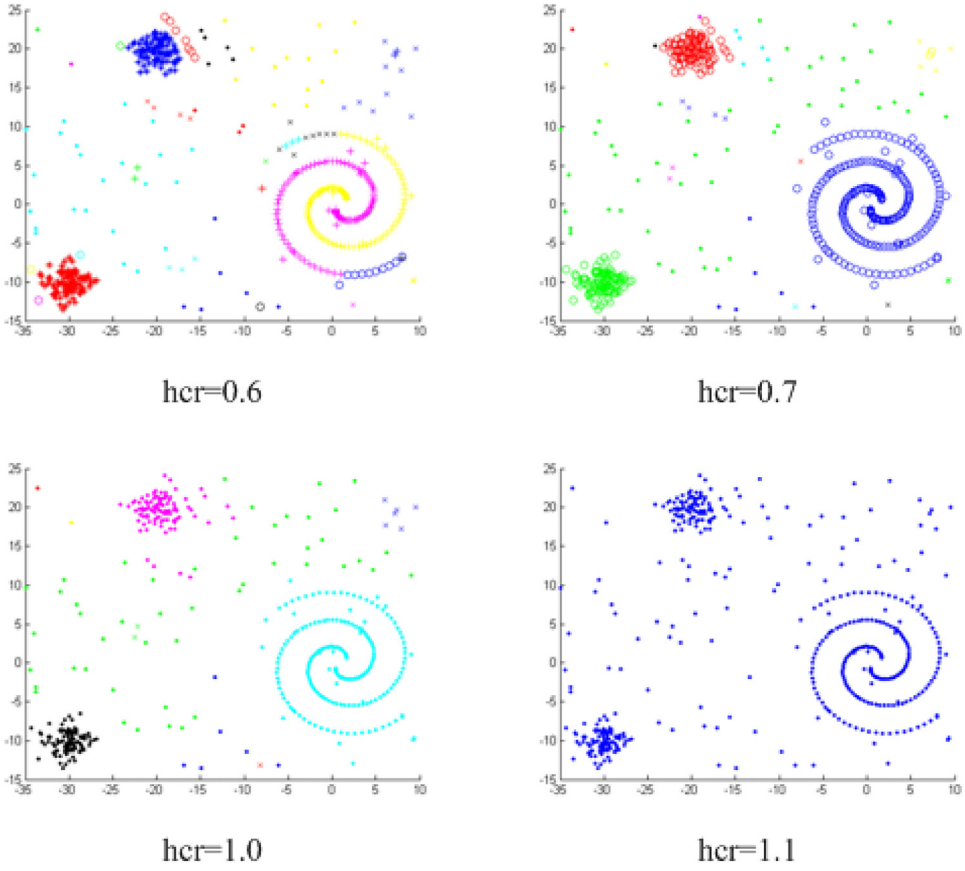


**Fig. 4.** Clusters obtained by our method for a composite point set, consisting of multiple types of clusters, with different values of the homogeneity criterion $h_{cr}$.

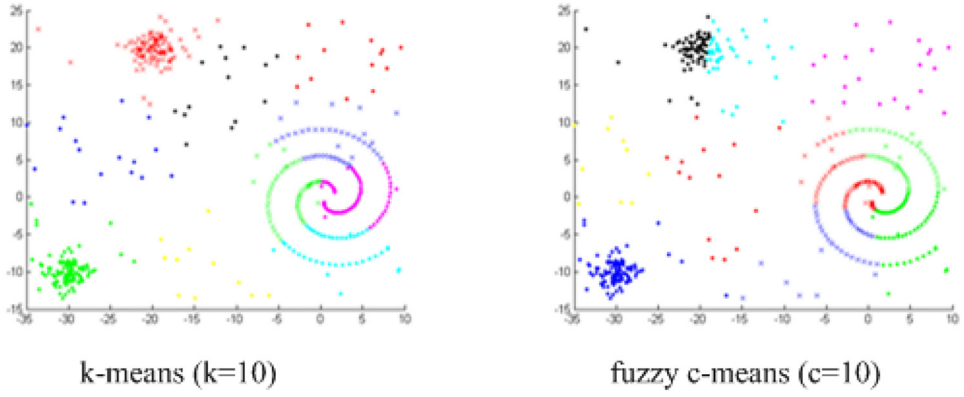


**Fig. 5.** Clusters obtained for the composite point set of Fig. 4 by *k*-means and fuzzy *c*-means methods.

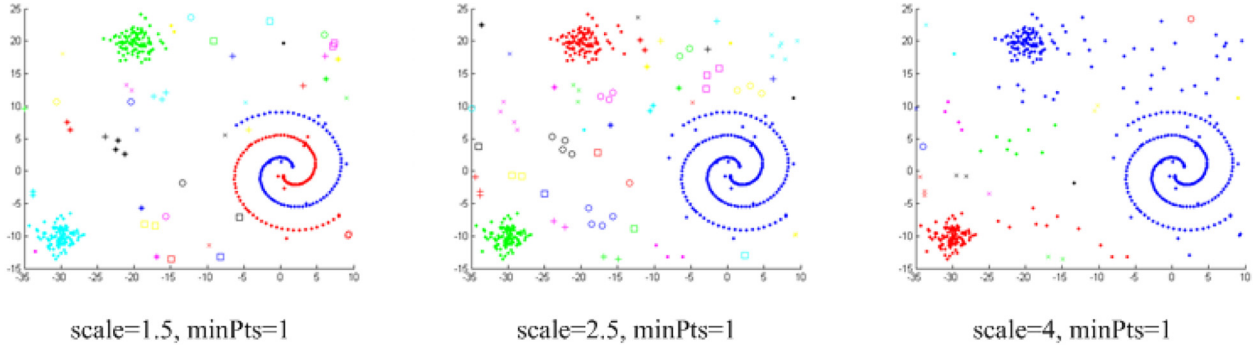


**Fig. 6.** Clusters obtained for the composite point set of Fig. 4 by DBSCAN. In the left-most figure, the number of clusters is very large, and therefore some of the clusters may not be displayed.

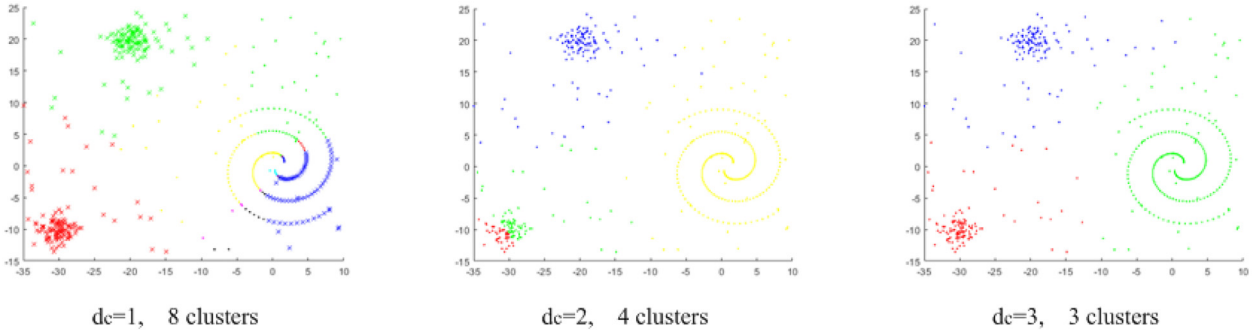


**Fig. 7.** Clusters obtained for the composite point set of Fig. 4 by the clustering method of Rodriguez and Laio [40], for various cutoff distance $d_c$ values.

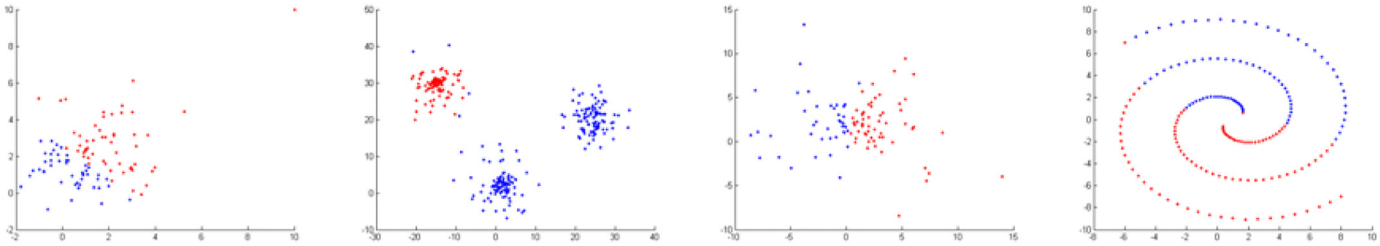

**Fig. 8.** Clusters obtained by Normalized-cut method, for bi-partitioning each point set shown in Fig. 3.

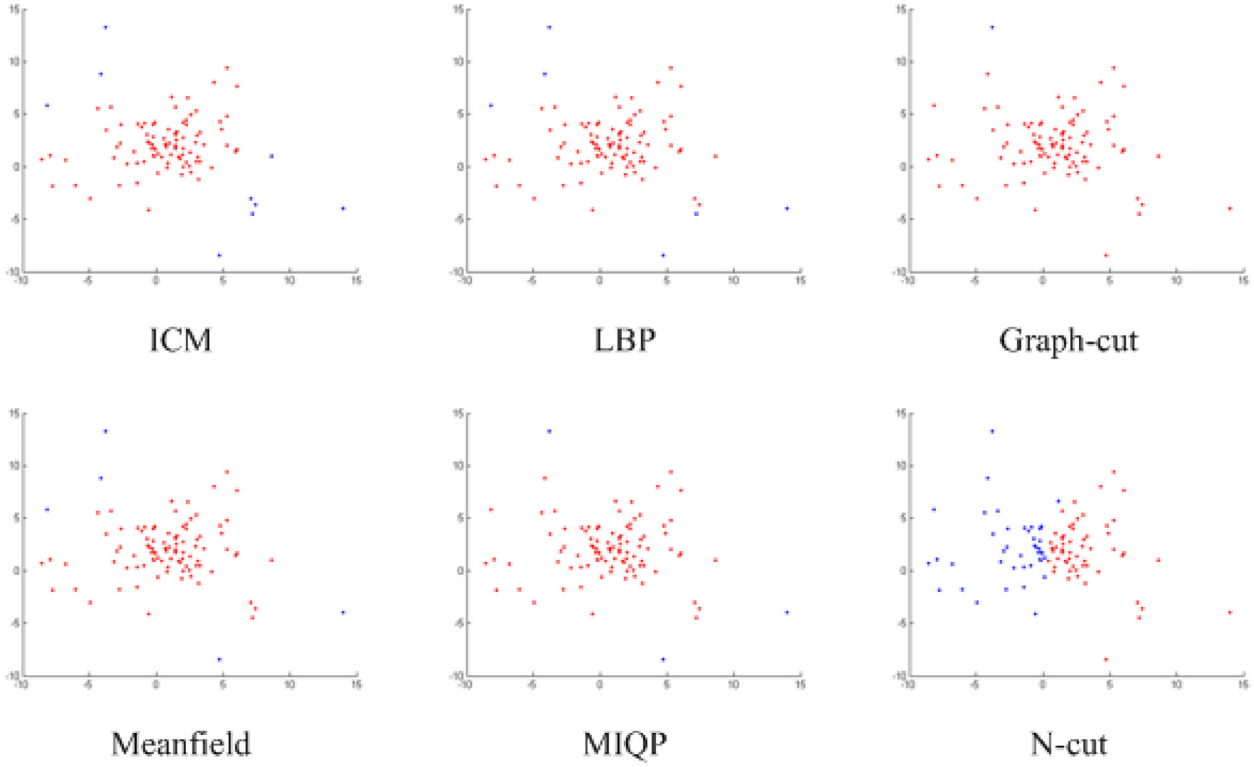


**Fig. 9.** Clustering results for various energy optimization algorithms, and Normalized-cut (N-cut) result is also shown for comparison.

and the background scatter points are also (roughly) classified as one cluster even though they are spread across the space, with other clusters being "embedded" within. When the homogeneity criterion $h_{cr}$ is made even smaller, some clusters further break into multiple clusters according to their density variations; for example, the spiral are divided into multiple clusters for each of its two branches.

In Fig. 5, we also compare our results with those obtained by *k*-means [20] and fuzzy *c*-means clustering [37], where *k*-means is in fact one of the most widely used clustering algorithms (using Matlab standard implementation). Both of these methods require the cluster number as the input, we set it as 10, which is the cluster number we obtain for the case $h_{cr} = 1.0$ using our method (Fig. 4). We see that both *k*-means and fuzzy c-means methods do not work well for this case as the spiral and the background scatter points are divided into visually non-intuitive clusters.

In Fig. 6, we compare our results with DBSCAN [25] (using the implementation of Peter Kovesi [38]). DBSCAN is the most well-

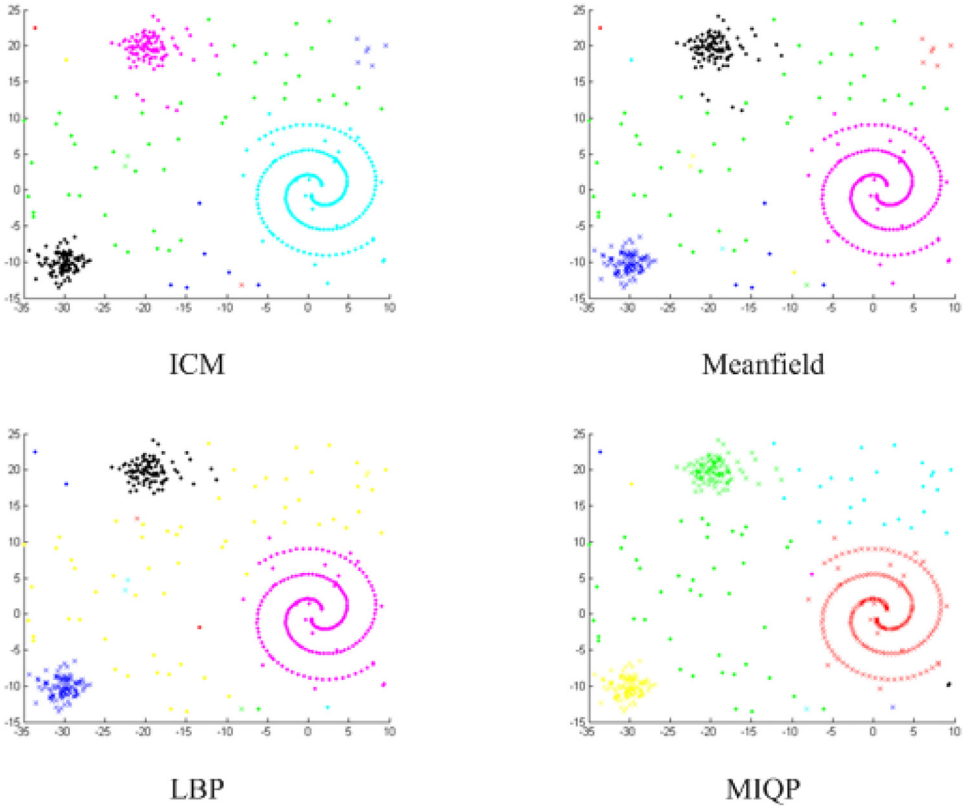


**Fig. 10.** Comparison of visual clustering results on the composite point set, by the energy optimization algorithms ICM, Meanfield, LBP and MIQP for homogeneity criterion $h_{cr} = 1.0$ .

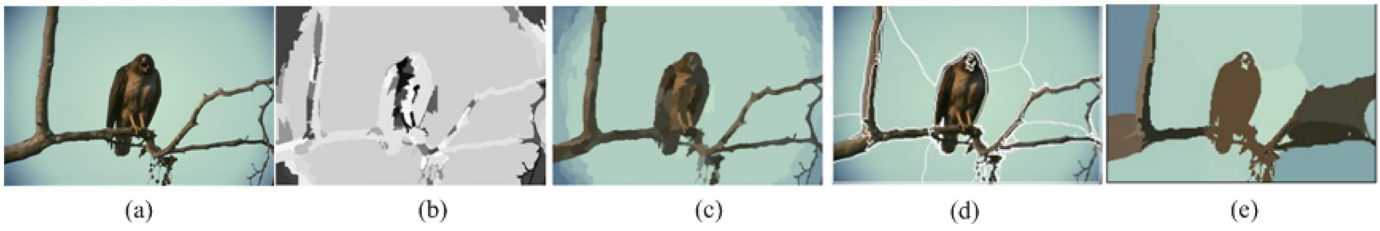


**Fig. 11.** Image segmentation by our method and Shapira [39]. (a) Original image. (b) Our segmentation shown in graylevel. (c) Our segmentation shown in simplified image. (d) Shapira [39]'s segmentation result. (e) Shapira [39]'s simplified image.

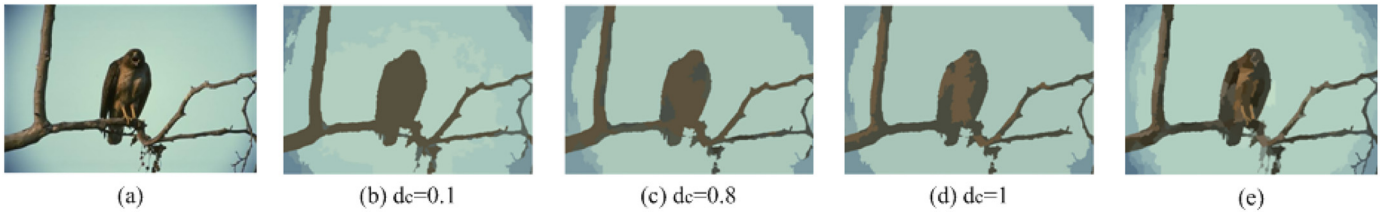


**Fig. 12.** Image segmentation by our method and Rodriguez and Laio [40]. (a) Original image. (b–d) The segmentation results of Rodriguez and Laio [40], in various $d_c$ values. (e) Our segmentation result.

known density-based clustering algorithm and also most-cited in scientific literature [1]. DBSCAN requires the input of a distance threshold (*scale*) and minimum number of points required to form a cluster (*minPts*). We compute the minimum distance between every pair of points in the dataset (in the range [0.0137, 5.8848]) and test various *scale* values within the above range. We set $minPts = 1$ so that no point is discarded in clustering. We can see that DBSCAN has the ability to differentiate the major clusters (spiral/branch and the dense point clouds), but it is unable to put the background scatter points (with similar local densities) into the same (or a few) clusters (they break into too many clusters). Also, the hierarchical merging of clusters when scale increases is not very reasonable either.

In Fig. 7, we compare our method with Rodriguez and Laio [40], which is also considered as the recent state-of-the-art clustering method. For each data point $i$, the method in [40] computes two quantities: its local density $\rho_i$ and its distance $\delta_i$ from points of higher density; in addition, it requires a tuning parameter called *cutoff distance*, $d_c$, for computing the local density. This algorithm has its basis in the assumptions that cluster centers are surrounded

**Fig. 13.** Our segmentation results on Berkeley dataset images, also the comparisons with the method in [40]. Top row in each panel (three-image column of the same image): original image. Middle row in each panel: our segmentation result in simplified image. Last row in each panel: the segmentation result obtained by Rodriguez and Laio [40].


**Fig. 14.** Detailed comparisons of our segmentation results with those obtained by Rodriguez and Laio [40] on Berkeley dataset images. For each row, the first image is the original image, the next three images are the segmentation results obtained by Rodriguez and Laio [40] with various $d_c$ values, and the last image is our segmentation result.

by neighbors with lower local density and at a relatively large distance from any points with a higher local density. Apparently, this assumption steps from the usual blob-like clusters, and does not suit the cases of some special shaped clusters, such as the spiral structure as illustrated in our examples. In fact, the method in [40] may have trouble in handling the case shown in Fig. 1, where the value of the tuning parameter *cutoff distance* $d_c$ is crucial for the performance. For example, at point A, a smaller $d_c$ and a larger $d_c$ (if it also includes the points of another cluster) could lead to totally different density results, and the computed local density may not even be meaningful. Hence the statement in [40] that "The algorithm is sensitive only to the relative magnitude of the densities in different points, ... , the results of the analysis are robust with respect to the choice of $d_c$" does not apply in this case.

As mentioned above, the method in [40] requires us input a parameter called *cutoff distance*, $d_c$; with the varying *cutoff distance*, the hierarchical clustering over the dataset can be obtained. Unfortunately, for a large range of the $d_c$ values, we notice that this method is unable to differentiate the spiral branches and separate them elegantly into two clusters; instead, it divides the spiral structure into several sector regions, and classifies the adjacent points on the spiral branches (the points on the same side), no matter which branch it belongs to, into the same cluster, and hence does not recognize the spiral structure at all (see the clustering result for $d_c = 1$ ). Also in the case of $d_c = 2$, the split of the point cloud at the left-bottom corner of the plot is unreasonable either. For the larger $d_c$ value $d_c = 3$, the clustering result is quite reasonable and the whole dataset is divided into three major

**Table 2**
Energy value obtained after optimization, and runtime for various energy optimization algorithms on the clustering problem shown in Fig. 9.

| | ICM | LBP | Graph-cut | Meanfield | MIQP |
|---|---|---|---|---|---|
| Energy value | −410.7787 | −410.6653 | −410.7464 | −412.3364 | −412.6643 |
| Runtime (s) | 0.6118 | 4.0016 | 0.0067 | 0.0600 | 0.5867 |

clusters, yet the spiral structure and the background scatter points are not differentiated in this case. From above experiments we see that among the tested clustering methods, those can recognize the two spiral branch structures only include our clustering method and DBSCAN.

#### 4.1.2. Comparison with Normalized-cut clustering method

For comparison with Normalized-cut, it is straightforward to replace our energy based density clustering module with the Normalized-cut module [22]; all other components in our framework — the neighborhood system, the local density computed at each node, and the weight matrix $W = \{w_{ij}\}$, remain unchanged. We perform the Normalized-cut clustering on the point sets shown in Fig. 3, but only to divide each point cloud into two clusters (Fig. 8). We can see the clustering results for most point sets are unreasonable, with points not clustered based on their local densities but split in the middle.

#### 4.1.3. Comparison of specific energy optimization algorithms within our scheme

There are many graphical model energy optimization algorithms, such as ICM (Iterative Conditional Mode) [11], LBP (Loopy belief Propagation) [12–14], Graph-cut [15,16], Mean field theory algorithm [17,18], as well as the integer programming algorithm (our problem in (2) is also an integer programming problem; we apply the integer programming toolbox Tomlab [19] (based on branch-and-cut) and name it "MIQP" (Mixed-Integer Quadratic Programming) in our experiment). We compare the performance of these specific energy optimization algorithms within our clustering framework, by simply replacing the energy optimization module in our scheme with various optimization algorithms above.

In order to show the energy value obtained after clustering conveniently, we first restrict the clustering to be one-level, that is, we bi-partition the original point set without further iterations. The experimental results are shown in Fig. 9 and Table 2. In addition, as a qualitative comparison, we also conduct a complete iterative clustering as that is done in Section 4.1.1 on the composite point set of Fig. 4, with various energy optimization algorithms (ICM, LBP, Meanfield and MIQP).

In Fig. 9 and Table 2, the clustering result of every optimization algorithm is somewhat different. While it is clear that the MIQP method achieves the lowest energy value with the reasonable clustering result, visually speaking the result obtained by ICM algorithm looks even better in that it separates the relatively dense core region and the relatively sparse boundary region of the point cloud. Again, the Normalized-cut method does not offer meaningful clustering for this point set.

Finally, we compare the clustering performance of applying various energy optimization algorithms (ICM, Meanfield, LBP and MIQP) within our scheme on the composite point set (Fig. 10). This time we perform a complete iterative clustering as we do in Section 4.1.1, with $h_{cr} = 1.0$ . All the four algorithms are able to segment out the three major clusters — the spiral and the two dense-core point clouds. However, while ICM, Meanfield and LBP grouping the background scatter points roughly into the same cluster, MIQP splits them into two parts via the "narrow neck" (yet it may seem more intuitive to classify all the background scatter points in one cluster). At the same time, MIQP's result on the background scatter points is cleanest — other methods always tend to have some outliers in this cluster. Other than MIQP, visually speaking LBP has the best performance in clustering the background scatter points. The runtimes of our method with various energy optimization algorithms are: ICM 2.3299s, Meanfield 0.4113s, LBP 68.3982s, MIQP 0.5903s. LBP is much slower than other algorithms at least in our implementation.

### 4.2. Experiments on real image task

#### 4.2.1. Image segmentation

We evaluate our clustering method on the real image task of image segmentation, and compare our results with some state-of-the-art method [39]. We use image superpixels as the basic unit for clustering. For the feature on each superpixel we use its average *Lab* color (the *L* channel is related with the brightness and should be removed); on the other hand, we think using pixel location XY as the feature (as in [39]) is unreasonable because some segments could have very elongated/irregular shape and farther apart of two points does not mean they are less likely to belong to the same segment. In addition, our method is totally unsupervised — we do not need to determine the bandwidth $r$ as needed in [39] (see Eq. (10) in [39], which would be critical for its performance). Our method does not need any user input parameters (say number of segments, bandwidth parameter, image size/contrast, etc.), one just inputs the image and obtains the segmentation. We show our segmentation result as a graylevel image and also as a simplified image. For the grayscale result, different grayscales denote different segments in the image; same graylevel denotes the same kind of segment (may be disconnected). So our output is beyond image segmentation, it can be considered as image recognition (segments with the same graylevel can be considered as the same type of object or scene). The simplified image means all the segments belonging to the same cluster will be assigned with the same color — the average color of this cluster. In order to compare with [39], we first test our method on the same images tested in (Fig. 8 of) [39] (images from Berkeley image dataset); the results are shown in Figs. 11–13. For the bird image in Fig. 11, we see our segmentation result is significantly better than that in [39]. In [39], the sky is split randomly although the colors of the split parts are the same (Fig. 11 (d)), while in our result the majority part of sky is classified as the same segment (Fig. 11 (b)(c)), and the outside region of sky is segmented out where the color difference is noticeable. Meanwhile our method keeps the image details much better (see the tree branches, compare Fig. 11 (c)(e)). Shapira [39] tends to split the large uniform region, it might be an intrinsic problem of Shapira [39]. Our runtimes are between 1.4 and 1.8s on an old laptop PC (Intel Core i3).

We also compare the segmentation performance of our method with the method of [40], with various cutoff distance $d_c$ values for the latter method (Fig. 12). The method in [40] can also give reasonable segmentation result in large. Yet it is hard to keep the details of the image in segmentation when compared with our method, such as the tree branches at the right-bottom corner of the image, as well as the details of the bird (say the color variation of the bird and the claws of the bird).

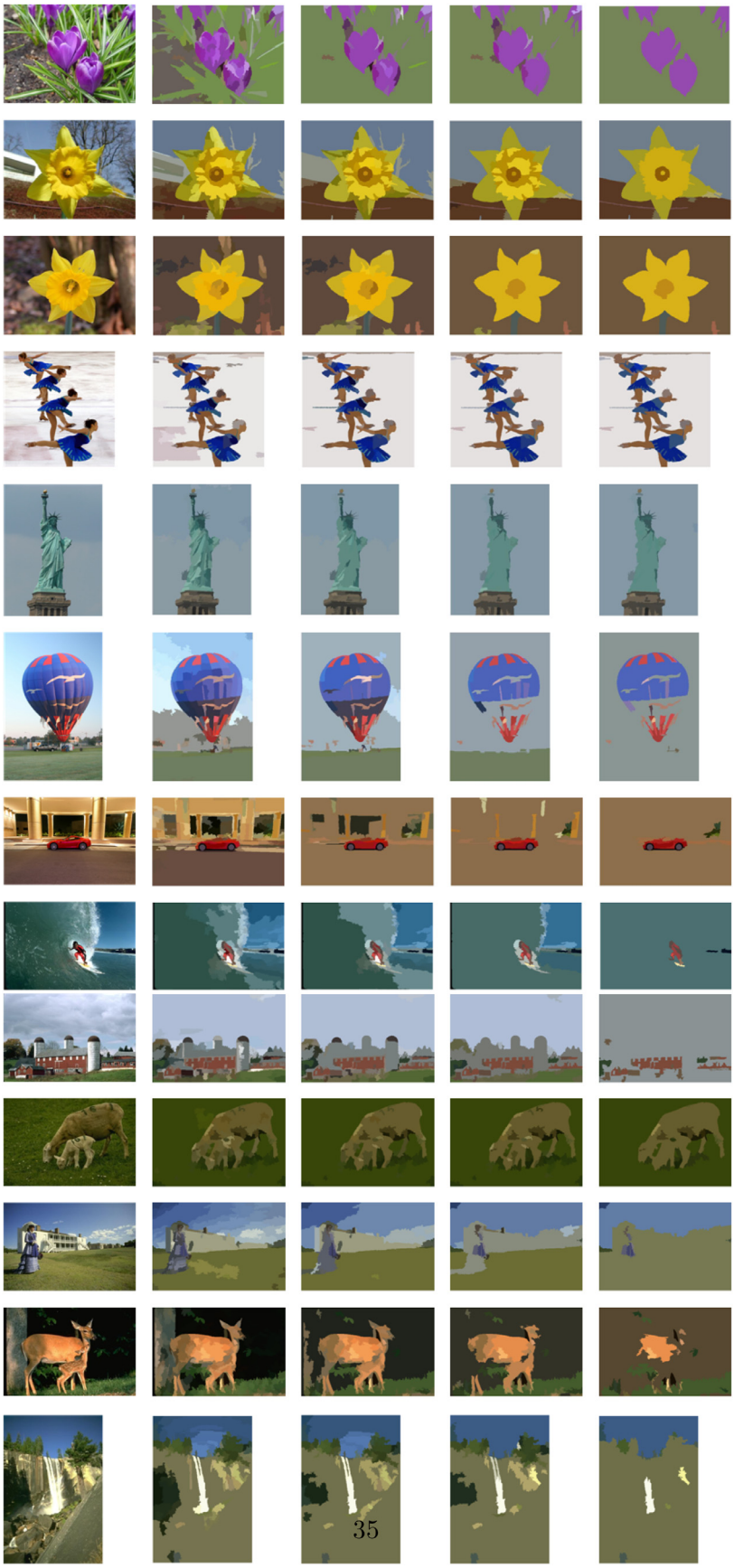


**Fig. 15.** The hierarchical image segmentation based on our clustering method. The first column is the original image taken from Oxford flower, iCoseg and Berkeley image datasets. The following columns are the hierarchical segmentation from fine to coarse.

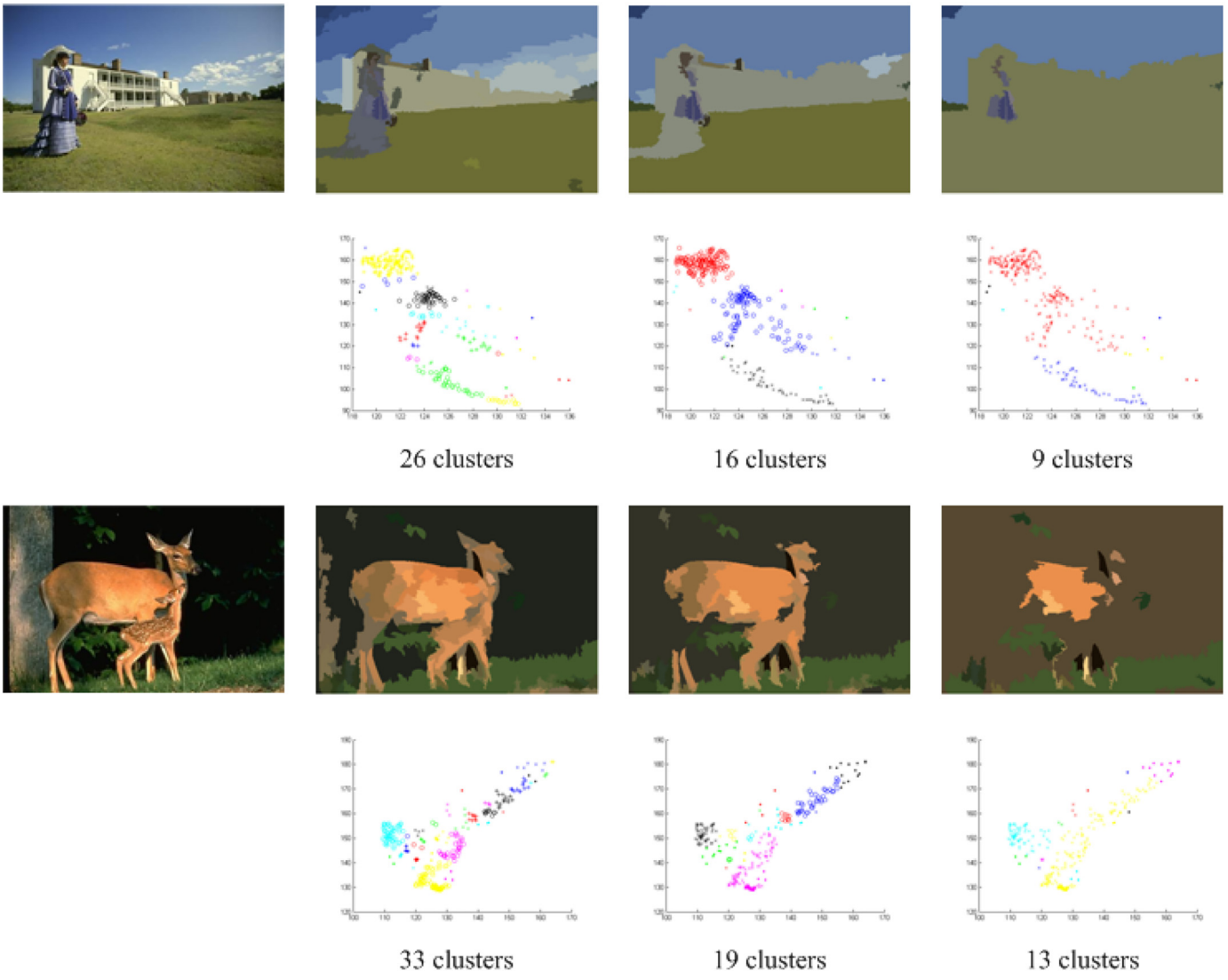


**Fig. 16.** The correspondence between the hierarchical clustering and the image segmentation. Odd rows: the original images and the hierarchical segmentation from fine to coarse. Even rows: the corresponding hierarchical clustering in the color space.

We also test our method on some other images in Berkeley dataset (Fig. 13). Meanwhile, we compare our image segmentation results with those obtained by the recent state-of-the-art clustering method [40] (Fig. 13). In Fig. 13, for each image case (the column with the three same images), the top one is the original image, the second one is our segmentation result, whereas the third one is the segmentation result obtained by Rodriguez and Laio [40]. Before making comparison, we see our segmentation results are meaningful and reasonable for the given images; especially, the details of the images can be kept well. In comparison with the method in [40], we see the segmentation performances of the two methods are comparable overall, both segmentations can reasonably explain the image under the consideration. Yet our method can keep the image details much better compared with the method in [40], which will be illustrated in Fig. 14.

In Fig. 14, we choose several images whose segmentation results are more different between the two methods, and try various parameter values, i.e. the cutoff distance $d_c$ values, for the clustering method in [40]. We see that even with various $d_c$ values for clustering, the method in [40] could not capture some image details in the segmentation. The small patches of distinct colors cannot be preserved in their method (for example in Row 1 and Row 3 of Fig. 14, the distinct colors of people's shirts); as well as the details in object shape and boundary (for example in Row 2 and Row 4 of Fig. 14, the details of the lady and the surfer); and particularly, the person behind the flag in Row 6 of Fig. 14.

#### *4.2.2. Hierarchical segmentation*

Since our clustering method can provide a hierarchy of clustering, by successively merging similar clusters, we can validate this capability via hierarchical image segmentation. Fig. 15 shows the results, where the first column is the original image and the rest columns show the hierarchical segmentation from fine to coarse. We can see the hierarchical segmentation is visually meaningful, which further validates the performance of our hierarchical clustering method. In Fig. 16, we further show the correspondence between the hierarchical clustering (in the color space) and the image segmentation.

## 5. Conclusion

In this work, we propose a clustering method based on physics-inspired graphical model energy optimization. The cluster-

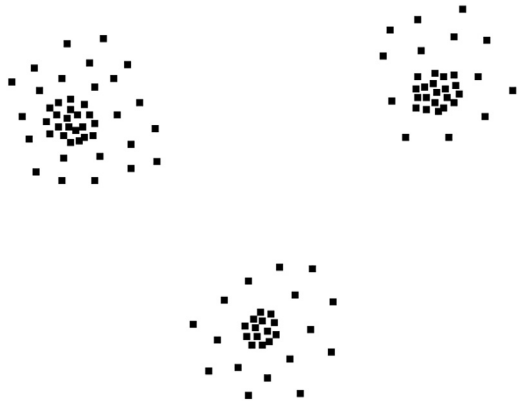

**Fig. 17.** The simple example showing that the homogeneous criterion is a target-related value.

ing method is unsupervised with no user input parameter needed. The energy formulation uses the Ising model as the basis, and we emphasize the using of the unary/data energy (in addition to the commonly used binary/pairwise energy) in the energy model, which represents the natural tendency of a node belonging to a certain cluster. We perform hierarchical clustering, where the value of a homogeneity criterion defines the hierarchy, with cluster density being more and more homogeneous lower down in the hierarchy. We also establish the connection between our energy optimization based clustering with the graph spectral clustering (such as Normalized-cut). We carry out experimental evaluations of our proposed method on both synthetic datasets and real-image tasks, and also compare our results with other clustering methods. In addition, we test the performance of various energy optimization algorithms within our clustering framework, such as ICM, LBP, Mean field theory algorithm, Graph-cut and the integer programming algorithm.

One of the major advantages of our method is parameter-free. Since our algorithm detects the local density naturally in the nearest neighborhood structure of the data set, we do not need to specify a scale or radius for detecting the local density as needed in DBSCAN; moreover, the local density becomes an attribute at each data point, we explore the unary (self) and binary (mutual) connections among those attributes naturally via a global energy optimization, hence we no longer need to rely on the local "count number" to make hard-thresholding. Also our method can handle the case well that the local density is evenly distributed within a point cloud with no obvious density gradient, while it might be challenging for the mode-seeking method such as mean-shift which requires the density gradient. Finally in our method more spatial information is explored via connected component analysis. The entire process requires no user parameter input. The only varying parameter, homogeneous criterion, is not a tuning-parameter, but a target-related value; various values may lead to reasonable clustering best suited for different purposes. A simple example is shown in Fig. 17, where we see that each point cloud pattern has a dense core and a sparse halo region. We may wish to cluster the data points into three point cloud patterns, including both the dense core and sparse halo region around it; or we may want to separate the core and halo regions apart, since the halo region may be the noise that we want to remove (in that case we have six clusters). The best clustering depends on the target we aim for, and the density fluctuation we can accept for our purpose. Both may be meaningful for certain problem. The algorithm can also provide a hierarchy of clustering according to different homogeneous criterion value, each may best suit a particular purpose. In practice we notice a default value (e.g. 0.7) of the criterion can give reasonable clustering in most cases.

Our method relies on the density property of each data point, therefore it is quite sensitive to density variations of the data space and sometimes it tends to split the same data point cloud into multiple regions. Of course it can be eliminated by adjusting the homogeneity criterion, i.e. allowing more density fluctuation within one cluster; but still sometimes the same homogeneity criterion level does not lead to the perfect clustering everywhere within the data space (e.g. in order to differentiate two clusters similar in density, you may over-split another cluster). In the future we may even consider an adaptive homogeneity criterion at various regions in data space.

Another future direction we would like to explore is the balance between the local density and spatial adjacency information in clustering. The early clustering methods rely heavily on the spatial adjacency among the data points, therefore they tend to create isotropic clusters and have difficulties in discovering the clusters of irregular shape. Modern methods tend to rely more on the local density property of the data points, therefore the clusters can be of any shape; yet the spatial information still needs to play an important role in clustering, such as in terms of the connected component analysis. Whether there are other ways to take the spatial information into account for clustering, or another scheme to combine the density information and spatial information elegantly together in clustering, is something we could explore in the future.

## Acknowledgments

The author would like to give the special thanks to Prof. Mark Hasegawa-Johnson at University of Illinois at Urbana-Champaign, for his kind and detailed editing of the paper content in many sections. The same thanks go for the anonymous reviewer as well. We also want to thank Mr. Marcus Edvall at Tomlab Optimization for the continuous support on the software license to facilitate our research. The support of the Office of Naval Research under grant N00014-16-1-2314 is gratefully acknowledged.

## References

[1] D. Xu, Y. Tian, A comprehensive survey of clustering algorithms, Ann. Data. Sci. 2 (2) (2015).
[2] A. K. Jain, M. N. Murty, P. J. Flynn, Data clustering: a review, ACM Comput. Surv. (CSUR) 31 (3) (1999).
[3] A.K. Jain, Data clustering: 50 years beyond k-means, Pattern Recognit. Lett. 31 (2010).
[4] P. Berkhin, Survey of clustering data mining techniques, book chapter in Grouping Multidimensional Data, pp 25–71, Springer, 2002.
[5] K. Rose, E. Gurewitz, G.C. Fox, Statistical mechanics and phase transitions in clustering, Phys. Rev. Lett. 65 (8) (1990).
[6] M. Blatt, S. Wiseman, E. Domany, Superparamagnetic clustering of data, Phys. Rev. Lett. 76 (18) (1996).
[7] A. Noack, Energy-based clustering of graphs with nonuniform degrees, book chapter in Graph Drawing, Volume 3843 of the series Lecture Notes in Computer Science, pp 309–320, 2005.
[8] A. Noack, Energy models for graph clustering, J. Graph Algorithms Appl. 11 (2) (2007).
[9] Y. Fu, P.W. Anderson, Application of statistical mechanics to NP-complete problems in combinatorial optimization, J. Phys. A 19 (1986).
[10] R. Albert, A.-L. Barabasi, Statistical mechanics of complex networks, Rev. Mod. Phys. 74 (2002).
[11] J.E. Besag, On the statistical analysis of dirty pictures, J. R. Stat. Soc. Ser. B 48 (3) (1986).
[12] J. Pearl, Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference (2nd ed.), Morgan Kaufmann, 1988.
[13] P.F. Felzenszwalb, Efficient belief propagation for early vision, In CVPR, 2004.
[14] J. Yedidia, W. Freeman, Y. Weiss, Generalized belief propagation, Advances in Neural Information Processing Systems, 2000.
[15] Y. Boykov, O. Veksler, R. Zabih, Fast approximate energy minimization via graph cuts, PAMI 23 (11) (2001).
[16] V. Kolmogorov, R. Zabih, What energy functions can be minimized via graph cuts, PAMI 26 (2) (2004).
[17] M. Saito, T. Okatani, K. Deguchi, Application of the mean field methods to MRF optimization in computer vision, CVPR, 2012.

[18] H.J. Kappen, W.J. Wiegerinck, Mean field theory for graphical models, In Advanced Mean Field Methods: Theory and Practice, chapter 4, The MIT Press, 2001.
[19] Tomlab optimization website, http://tomopt.com/tomlab/.
[20] J. MacQueen, Some methods for classification and analysis of multivariate observations, in: Proc. Fifth Berkeley Symp Math Stat Probab, 1, 1967.
[21] D. Comaniciu, P. Meer, Mean shift: a robust approach toward feature space analysis, IEEE Trans. Pattern Anal. Mach. Intell. 24 (2002).
[22] J. Shi, J. Malik, Normalized cuts and image segmentation, IEEE Trans. Pattern Anal. Mach. Intell. 22 (2000).
[23] E. Ising, Beitrag zur theorie des ferromagnetismus, Z. Phys. 31 (1925).
[24] C. Wang, N. Komodakis, N. Paragios, Markov random field modeling, inference & learning in computer vision & image understanding: a survey, Comput. Vision Image Understanding 117 (2013).
[25] M. Ester, H. Kriegel, J. Sander, X. Xu, A density-based algorithm for discovering clusters in large spatial databases with noise, in: In: Proceedings of the second ACM SIGKDD international conference on knowledge discovery and data mining, 1996.
[26] H. Yang, N. Ahuja, Automatic segmentation of granular objects in images: combining local density clustering and gradient-barrier watershed, Pattern Recognit. 47 (2014).
[27] N. Ahuja, Dot pattern processing using voronoi neighborhoods, PAMI 4 (3) (1982).
[28] R.B. Potts, Some generalized order-disorder transformations, Math. Proc. 48 (1) (1952).
[29] B. Delaunay, Bulletin de l'acadmie des sciences de l'URSS, Classe des sciences mathmatiques et naturelles 6 (1934).
[30] H. Samet, M. Tamminen, Efficient component labeling of images of arbitrary dimension represented by linear bintrees, IEEE Trans. Pattern Anal. Mach. Intell. 10 (1988).
[31] L. He, Y. Chao, K. Suzuki, Arun-based two-scan labeling algorithm, IEEE Trans. Image Process. 17 (5) (2008).
[32] M.B. Dillencourt, H. Samet, M. Tamminen, A general approach to connected- component labeling for arbitrary image representations, J. ACM 39 (2) (1992).
[33] S. Shetty, N. Ahuja, A uniformity criterion and algorithm for data clustering, ICPR, 2008.
[34] F. Chung, Spectral graph theory, Am. Math. Soc. (1997).
[35] A. Pothen, H.D. Simon, K.P. Liou, Partitioning sparse matrices with eigenvectors of graphs, SIAM J. Matrix Anal. Appl. 11 (1990).
[36] G.H. Golub, C.F.V. Loan, Matrix Computations, John Hopkins Press, 1989.
[37] S. Chiu, Fussy model identification based on clustering estimation, J. Intell. Fussy Syst. 2 (3) (1994).
[38] Peter kovesi's implementation of DBSCAN clustering algorithm, 2013, http://www.peterkovesi.com/matlabfns/Misc/dbscan.m.
[39] L. Shapira, et al., Mode-detection via median-shift, ICCV, 2009.
[40] A. Rodriguez, A. Laio, Clustering by fast search and find of density peaks, Science 344 (6191) (2014).
[41] P. Franti, O. Virmajoki, Iterative shrinking method for clustering problems, Pattern Recognit. 39 (2006).
[42] H. Chang, D.Y. Yeung, Robust path-based spectral clustering, Pattern Recognit. 41 (2008).
[43] P. Franti, O. Virmajoki, V. Hautamaki, Fast agglomerative clustering using a k-nearest neighbor graph, IEEE Trans. Pattern Anal. Mach. Intell. 28 (11) (2006).
[44] Y. Zhu, K. Ming Ting, M. Carman, Density-ratio based clustering for discovering clusters with varying densities, Pattern Recognit. 60 (2016).
[45] L. Bai, X. Cheng, J. Liang, H. Shen, Y. Guo, Fast density clustering strategies based on the k-means algorithm, Pattern Recognit. 71 (2017).
[46] J.N. Myhre, K.O. Mikalsen, S. Lokse, R. Jenssen, Robust clustering using a kNN mode seeking ensemble, Pattern Recognit. 76 (2018).
[47] D. Huang, J. Lai, C.-D. Wang, Ensemble clustering using factor graph, Pattern Recognit. 50 (2016).
[48] F. Saki, N. Kehtarnavaz, Online frame-based clustering with unknown number of clusters, Pattern Recognit. 57 (2016).
[49] Y. Qin, Z.L. Yu, C.-D. Wang, Z. Gu, Y. Li, A novel clustering method based on hybrid k-nearest-neighbor graph, Pattern Recognit. 74 (2018).

**Huiguang Yang** received his Ph.D. from Electrical and Computer Engineering, University of Illinois, Urbana-Champaign. He received his B.S. in Environmental Engineering, Tsinghua University, Beijing, China, and received his M.S. in Atmospheric Sciences, University of Illinois, Urbana-Champaign. His research in Electrical and Computer Engineering focuses on image processing and computer vision, such as image segmentation and object recognition. He is now a researcher at Samsung Research Institute China, Xian (SRCX).

**Narendra Ahuja** received his Ph.D. from the University of Maryland, College Park, in 1979. He is Donald Biggar Willet Professor in Department of Electrical and Computer Engineering, University of Illinois, Urbana-Champaign. His fields of professional interest are next generation cameras, 3D computer vision, video analysis, image analysis, pattern recognition, human computer interaction, image processing, image synthesis, and robotics.