Iterated Watersheds, A Connected Variation of K-Means for Clustering GIS Data

Sampriti Soor, *Student Member, IEEE,* Aditya Challa, *Student Member, IEEE,* Sravan Danda, *Student Member, IEEE,* B. S. Daya Sagar, *Senior Member, IEEE* and Laurent Najman, *Senior Member, IEEE*

Abstract—In this article, we propose a novel algorithm to obtain a solution to the clustering problem with an additional constraint of connectivity. This is achieved by suitably modifying K-Means algorithm to include connectivity constraints. The modified algorithm involves repeated application of watershed transform, and hence is referred to as *iterated watersheds*. Detailed analysis of the algorithm is performed using toy examples. Iterated watersheds is compared with several image segmentation algorithms. It has been shown that iterated watersheds performs better than methods such as spectral clustering, isoperimetric partitioning, and K-Means on various measures. To illustrate the applicability of iterated watersheds - a simple problem of placing emergency stations and suitable cost function is considered. Using real world road networks of various cities, iterated watersheds is compared with K-Means and greedy K-center methods. It is observed that iterated watersheds result in 4 - 66 percent improvement over K-Means and in 31 - 72 percent improvement over Greedy K-Centers in experiments on road networks of various cities.

Index Terms—Graph Clustering, K-Means, E-governance, Watersheds,

1 Introduction

In digital age new approaches for effective and efficient governance strategies can be established by exploiting the vast computing and data resources at our disposal. In several cases, the problem of efficient governance translates to finding a solution to an optimization problem. A typical example is where several cases are framed in terms of clustering problem. The problem of clustering is - Given a set of objects, partition the set into clusters such that objects belonging to the same cluster are similar to each other, while objects belonging to different clusters are dissimilar to each other. There exists several possible solutions to the clustering problem (See [1], [2] for a comprehensive list of methods). The most commonly used method is that of *K-Means* [3]. This operates by minimizing the sum of dissimilarities within a cluster. By assuming that each object to be clustered is denoted by a data point, let $V = \{x_i\}$ denote a set of all data points. Also assume that the dataset must be clustered into k clusters. K-Means solves the following optimization problem.

$$\begin{aligned} \text{minimize} \quad & \sum_{x \in V} d(x, M(x)) \\ \text{subject to} \quad & |\{M(x) | x \in V\}| = k \end{aligned} \tag{1}$$

where, $M(x) \in \{c_1, c_2, \cdots, c_k\}$ denotes the center of the cluster to which x belongs, $|S|$ indicates the cardinality of the set S, and $d(.,.)$ denotes a dissimilarity measure. The problem of obtaining a global minima to (1) is in general NP-Hard. In particular, it was shown in [4] that the optimization problem in (1) is NP-Hard, if the data points are assumed to be in the euclidean space and dissimilarity measure is taken to be the euclidean distance. K-Means provides an algorithm to approximate the minima.

In several practical applications, a solution to the problem of clustering is required to have additional properties. One such property is that of "connectivity". For example, consider the problem of image segmentation, which is equivalent to the clustering problem. It is usually the case that a cluster (object or a part of it) in an image is expected to be "spatially connected". In Fig. 1(a), the object is a loop at the center of the image. K-Means algorithm does not necessarily preserve connectivity as shown in Fig. 1(b). A much better clustering in this case is presented in Fig.1(c).

In this article, we describe an algorithm similar to that of K-Means, which preserves connectivity. To allow for mathematical formalization, a graph structure is assumed. That is, we assume that apart from the set V, there exists a set $E \subseteq V \times V$, which gives the adjacency relation between two points. The elements of the set E are referred to as edges. A subset $S \subset V$ is said to be connected if, for any two points $x, y \in S$, one can reach y from x with a series of edges. The formal definitions are given in Section 2. Adapting the optimization problem in (1) to include connectivity of the resultant clusters, we have the optimization problem as in (2).

$$\begin{aligned} \text{minimize} \quad & \sum_{x \in V} d(x, M(x)) \\ \text{subject to} \quad & |\{M(x) | x \in V\}| = k \\ & \{x | M(x) = c_i\} \text{ is connected for all } i \end{aligned} \tag{2}$$

The aim of this article is to propose an efficient algorithm to approximate the minima of optimization problem (2), and analyze it on several cases. Such an algorithm could have several applications. Several problems related to e-governance and e-administration can be phrased as the

Sampriti Soor, Aditya Challa, Sravan Danda and B.S.Daya Sagar are with the Systems Science and Informatics Unit, Indian Statistical Institute, Bangalore, Karnataka, India 560059. E-mail: sampreetiworkid@gmail.com, aditya.challa.20@gmail.com, sravan8809@gmail.com, bsdsagar@isibang.ac.in
Laurent Najman is with Université Paris-Est, LIGM (UMR 8049), CNRS, ENPC, ESIEE Paris, UPEMLV, F-93162 Noisy-le-Grand, France. E-mail: laurent.najman@esiee.fr

(a) (b) (c)

Fig. 1. Example Illustrating that K-Means does not preserve connectivity. (a) Example grey scale image. (b) Clusters obtained using K-Means on the greyscale values of (a). (c) Clusters obtained using adapted K-Means algorithm as proposed in this article. This figure is also shown in [5].

optimization problem in (2). The following describes a few such problems -

- **Zonation Problem:** One of the most common problems faced during governance is that of splitting the entire set into zones. For instance consider the railway zonation problem, where stations are grouped under various zones for easy administration. In several of these problems, it is also the case that one requires connectivity of the zones. This can be solved using the optimization problem in (2). For the railway zonation problem, each railway station can be thought of a vertex and two adjacent stations are connected by an edge. Then a solution to the optimization problem in (2) gives a zonation in which zones are connected.
- **River Linking:** Another problem of importance is that of the river linking. Since the cost of linking rivers is huge, one can use the optimization problem (2) to identify which rivers to link to obtain the least cost.
- **Facility Allocation:** One of the common problems occurred is that facility allocation - Identify the cities in which the facilities must be located to minimize the costs of running the facilities. This problem is NP-Hard. Optimization problem in (2) can be used to find an approximate solution to this.
- **Analysis of Geo-Spatial data:** Apart from the above problems, a solution to (2) can be used to visualize geo spatial data as well. For instance, assume n entities trading among each other. Taking the vertex set to be the entities and edges among the entities if the trade is non-negligible, the optimization problem in (2) can be used to cluster the entities to visualize closely knit blocs with high trade among themselves.

In this article, we describe an algorithm that imitates the K-Means algorithm while preserving connectivity. In Section 2, we review the relevant literature required for the rest of the article. In Section 3, the proposed algorithm, which we call *iterated watersheds*, is presented. We provide a detailed analysis of the algorithm using various toy examples as well as using the problem of image segmentation in Section 4 . In Section 5, we then apply this algorithm to identify the 'ideal' locations for emergency facilities using road network datasets taken from [6].

A related work to the one presented in this article is given in [7], where the author used topographical distances to classify the pixels of an image. The work presented here can be seen as a generalization of [7] to general edge weighted graphs with any type of monotonically increasing distance functions, and with a much wider range of applications such as road networks.

2 Review of Relevant Literature

The proposed algorithm in next section combines concepts from K-Means as well as watersheds from Mathematical Morphology (MM). Accordingly, we review the relevant concepts in this section.

2.1 K-Means

K-Means is perhaps the most widely known classic algorithm for clustering [3]. See [2] for more details. Firstly, pick k random points as centers from V, k being the required number of clusters. The algorithm consists of repeating the following steps iteratively until convergence.

(a) Assign each of the points in V to one of the k centers based on the measure $d(.,.)$. This gives a partition of V.
(b) For each of the classes in the partition obtained from previous step, calculate the new center of this class. Thus, we have k new centers. Go to (a) using the new k centers.

There exists several explanations to explain why the above algorithm works. One of the explanations involves the expectation-maximization (EM) algorithm [8]. Step (a) can be interpreted as - Given the centers, identify the labelling which minimizes the cost in problem (1). This is known as a *maximization step*. Step (b) involves identifying the centers given the labelling of the points, also known as *expectation step*. Thus, K-Means closely follows the EM algorithm, and is identical to it if the data follows Gaussian distribution. In the next section, this similarity is exploited to propose an algorithm to get approximate optimum of problem (2).

Remark: K-Means algorithm is also used in other clustering methods as well. Spectral clustering uses K-Means as the final step for labelling [9]. Spectral clustering has been shown to be similar to *Kernel K-Means* (K-Means which uses a kernel to calculate the distance) [10]. Thus, in this article we compare our method with spectral clustering and classic K-Means.

2.2 Watersheds and IFT

Assume that $G = (V, E, W)$ denotes an edge-weighted graph, where V denotes the set of vertices, $E \subseteq V \times V$, denotes the set of edges, and $W : E \to \mathbb{R}^+$ denotes the edge weight assigned to each edge. These edge weights can be obtained as a restriction of the measure $d(.,.)$. Let $e = (e_0, e_1)$ denote an edge. Then,

$$W(\langle e_0, e_1 \rangle) = d(e_0, e_1) \tag{3}$$

A path $\pi = \langle x = x_0, x_1, x_2, \cdots, x_n = y \rangle$ between two vertices x and y is a sequence of edges - $(x_0, x_1), (x_1, x_2) \cdots$. Two vertices x and y in V are said to be connected if there

exists a path between them. A subset of vertices, $X \subseteq V$, is said to be connected if any two points in X are connected. If the whole set of vertices, V, is connected, then the graph is said to be connected.

Watersheds is a morphological transformation which can be used to segment images [11]. It uses the principle of steepest descent to assign each of the vertices to a unique minima. And each vertex is assigned a cluster based on the minima. That is, vertices which are assigned to the same minima belong to the same cluster. This concept has been adapted to edge weighted graphs in [12], [13], where quasi-linear algorithm was proposed to calculate the watersheds.

Efficient algorithm for computing watersheds is proposed in [14], known as the *Image Foresting Transform* (IFT). Intuitively, the algorithm proceeds by computing the shortest paths between any two points in V. To measure the cost of a path π, assume that there exists a function f, *path-cost function*, that assigns a cost for each path π. Examples of such functions are the additive cost function

$$f_{sum}(\pi.\langle s,t\rangle) = f_{sum}(\pi) + W(\langle s,t\rangle) \tag{4}$$

or the *pass value function* given by

$$f_{max}(\pi.\langle s,t\rangle) = \max\{f_{max}(\pi), W(\langle s,t\rangle)\} \tag{5}$$

Here $\pi.\langle s,t\rangle$ indicates the concatenated path obtained by adding the edge (s,t) to the path π.

Remark: For the IFT algorithm to return shortest paths, it is required that the path cost function f be monotonically incremental (MI), that is (a) $f(\pi.\langle s,t\rangle) \geq f(\pi)$ and (b) $f(\pi_1) \geq f(\pi_2)$ if and only if $f(\pi_1.\langle s,t\rangle) \geq f(\pi_2.\langle s,t\rangle)$. The path cost functions in (4) and (5) both satisfy this criterion. Another possible path cost function is

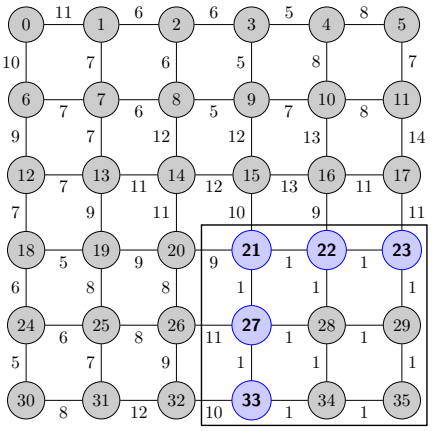
$$f_{comb}(\pi) = (f_{max}(\pi), f_{sum}(\pi)) \tag{6}$$

where $f_{comb}(\pi_1) < f_{comb}(\pi_2)$ if (i) $f_{max}(\pi_1) < f_{max}(\pi_2)$ or (ii) $f_{max}(\pi_1) = f_{max}(\pi_2)$ and $f_{sum}(\pi_1) < f_{sum}(\pi_2)$. This is also known as *dictionary ordering*, which can be useful in certain situations as well.

In the rest of the article, additive cost function is used unless mentioned otherwise. We review the IFT algorithm below for completeness. This is later used to efficiently partition the graph given the seeds.

Input: Edge weighted graph $G = (V, E, W)$, set of source points S, and initial labels of the source points.
Output: Labelling of all the vertices in V, L.

1: For all vertices $x \in V$, initialize - (i) $L(x) = Null$ if x is not labelled else, $L(x)$ is the label. (ii) Cost Map, $\mathcal{C}$ such that $\mathcal{C}(x) = 0$ for all x labelled and ∞ otherwise.
2: Initialize a priority Queue Q and insert all x to Q for which $\mathcal{C}(x) < \infty$.
3: **while** Q is not empty **do**
4: Pick the element with least cost from Q, say s_0.
5: **for** each $s_1 \sim s_0$ (adjacent to), and $\mathcal{C}(s_1) > \mathcal{C}(s_0)$ **do**
6: Compute cost $\mathcal{C}'$ of the path $\pi(s_0).\langle s_0, s_1\rangle$, where $\pi(s_0)$ denotes the optimal path to s_0.
7: **if** $\mathcal{C}' < \mathcal{C}(s_1)$ **then**
8: Update $\mathcal{C}(s_1)$, $L(s_1)$ and value of s_1 in Q.
9: **end if**
10: **end for**
11: **end while**
12: **return** L



(a)

Fig. 2. Example illustrating the boundary vertices. Vertices are shown in grey. Edge weights are as given near the corresponding edge. The bounding box indicates the connected subgraph. Vertices in blue-bold correspond to the boundary vertices for this subgraph.

2.3 Calculating the center of the cluster

One of the steps in K-Means algorithm is to calculate the center of the cluster (Expectation Step). There exists several methods to calculate the center depending on the domain of the data. A few such methods are discussed here.

1) If the data belongs to a space where it is possible to calculate averages, such as Euclidean space, then the center can be obtained by taking the averages. However, this center usually would not belong to the data. Hence, the closest point to the average of the data is taken as the center.
2) If the domain to which data belongs does not have any specific structure, then the center of the connected subgraph, C, can be defined as

$$x^* = \arg\min_{x \in C} \max_{y \in C} d(x,y) \tag{7}$$

The naive approach of computing the center of each cluster is the Floyd-Warshall algorithm [15]. Distances between all possible pairs of vertices is first calculated, and they are used to calculate the center. The cost of this operation is $\mathcal{O}(|C|^3)$. There also exists faster GPU versions of the algorithm as well [16].
3) Alternatively, one can use the distances between the boundary points and the remaining points to compute the center. This reduces the complexity significantly. We say a vertex x, belonging to a subgraph C, is a *boundary vertex* if it is adjacent to a vertex in $V \setminus C$. An illustration is given in Fig. 2.

3 Iterated Watersheds

Recall that the problem of clustering that preserves connectivity is framed as the following optimization problem. Assume that k clusters are required and the set $E \subseteq V \times V$ is given according to which the connectivity is defined.

$$\begin{aligned} \text{minimize} \quad & \sum_{x \in V} d(x, M(x)) \\ \text{subject to} \quad & |\{M(x) | x \in V\}| = k \\ & \{x | M(x) = c_i\} \text{ is connected for all } i \end{aligned} \tag{8}$$

where, $M(x) \in \{c_1, c_2, \cdots, c_k\}$. Here, one needs to find the centers c_i as well as the map M, which assigns each vertex in V to a center.

As a first step, we slightly modify the optimization problem in (8), by defining dissimilarity measure $d_W(.,.)$ as

$$d_W(x, y) = \min_{\pi \in \Pi(x,y)} f(\pi) \tag{9}$$

where $\Pi(x, y)$ denotes the set of all paths between x and y. The optimization problem is then rephrased as

$$\begin{aligned} \text{minimize} \quad & \sum_{x \in V} d_W(x, M(x)) \\ \text{subject to} \quad & |\{M(x) | x \in V\}| = k \\ & \{x | M(x) = c_i\} \text{ is connected for all } i \end{aligned} \tag{10}$$

The following reasons justify this modification:

1) We say that d_W is equivalent to d, if, given any two pairs of points (x_1, x_2) and (y_1, y_2), we have

$$d_W(x_1, x_2) \leq d_W(y_1, y_2) \Leftrightarrow d(x_1, x_2) \leq d(y_1, y_2) \tag{11}$$

In these cases, we have that solving optimization problem in (8) is equivalent to solving optimization problem in (10). Thus, under a relatively minor condition, we have that these optimization problems are equivalent. A straightforward instance of such an equivalence is that when using a complete graph and $d_W = d$.

2) In the most general case, it is seen that graphs constructed by using the k-nearest neighbors reflect the structure of the data. For instance, this property is exploited for dimensionality reduction in [17]. In such cases, one can make an argument for solving optimization problem in (10) is more suitable than solving the optimization problem in (8).

The algorithm to optimize the problem in (10), and to calculate the clusters is now described. We refer to this as the **Iterated Watersheds** algorithm.

Input: Edge weighted graph $G = (V, E, W)$, number of clusters k.

Output: Labelling $L : V \to \{1, 2, \cdots, k\}$

1: Pick k random vertices from V - $\{c_1, c_2, \cdots, c_k\}$.
2: Initialize $C_i \leftarrow \{c_i\}$ for all i.
3: **while** convergence is not reached **do**
4: Using IFT algorithm, assign each vertex x to its nearest center c_i (say). $C_i \leftarrow C_i \cup \{x\}$.
5: Compute the centers of each C_i.
6: **end while**

Intuitively, each of the two steps of the iterated watersheds algorithm is similar to expectation and maximization steps of the K-Means algorithm. Thus, the above algorithm can also be considered a variant of K-Means which preserves connectivity.

This algorithm has the following properties:

Property 1: At the end of each iteration, the clusters C_i are connected.

This is due to the fact that IFT algorithm calculates the distances via paths on the graphs, and hence each vertex is connected to the center it is assigned to. It follows that, two vertices assigned to the same center are hence connected.

Property 2: Given the centers $\{c_1, c_2, \cdots, c_k\}$, the IFT algorithm minimizes the cost

$$\sum_{x \in V} d_W(x, M(x)), \tag{12}$$

where $M(x)$ denotes the center to which x is assigned. This, once again, follows from the property of the IFT algorithm which assigns each vertex to its nearest center.

Relation with Kernel K-Means

A related technique to the algorithm above is the *Kernel K-Means* [10], where instead of calculating the distances $d(a, b)$ as in K-Means, it uses a transformation ϕ to calculate $d(\phi(a), \phi(b))$. Intuitively, Kernel K-Means distorts the distances, which allows the method to identify non-convex clusters. Iterated watersheds can also be intuitively seen as distorting the distance measurements by considering distance along the edges. Hence, iterated watersheds is closely related to kernel K-Means.

The main difference is, instead of using the transformation ϕ, the above method uses an edge weighted graph and the shortest distance on it to calculate the distance. Thus, iterated watersheds calculate the distances without using explicit kernels.

In [10], the authors mention that the kernel K-Means algorithm is similar to the spectral clustering [9], and by extension it can be seen that iterated watersheds is similar to spectral clustering as well. Thus, in this article we compare the results with spectral clustering techniques, along with related isoperimetric partitioning and K-Means technique.

The main advantage of iterated watersheds over spectral clustering is that - solving for eigenvectors can be numerically unstable in several cases. This does not matter for iterated watersheds.

4 Analysis and Illustrations

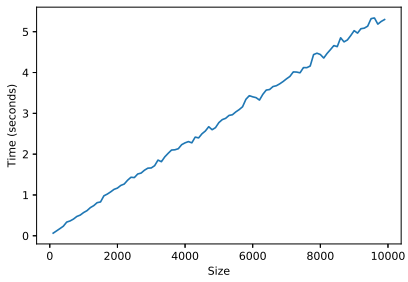
In this section we analyze and illustrate the behavior of the algorithm by using a few experiments.

Implementation Notes:

(i) The iterated watersheds algorithm is compared mainly with spectral clustering algorithm, since both can be identified as adaptation of K-Means.
(ii) As with K-Means, iterated watersheds must also be repeated with several different initializations to make sure that all objects are identified. Each result of the iterated watersheds presented here is taken from the best of 10 repetitions.
(iii) Since, several experiments are based on data in Euclidean space, the nearest vertex to the average is taken to be the new center.

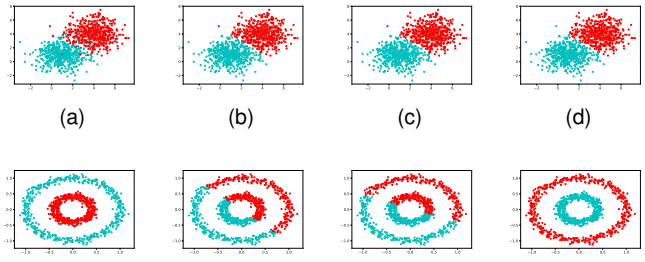
4.1 Time Complexity of Iterated Watersheds

Assuming that the number of clusters required is much lesser than the data size, $k << n$, step (4) of the algorithm takes linear time to label all the vertices. Time complexity of step (5) of the algorithm depends on the method of finding the centers. Using the nearest vertex to the average, step (5) takes linear time as well. This is verified empirically in Fig. 3.



(a)

Fig. 3. Example illustrating the time complexity. It can be seen that the iterated watersheds scales linearly with size. The dataset considered is "make_blobs" in sklearn [18] with number of centers = 2 and standard deviation = 1.25.



(a) (b) (c) (d)

(e) (f) (g) (h)

Fig. 4. Example to illustrate the working of iterated watersheds. (a) - (d) uses the make blobs dataset [18]. (a) shows the original data/classes. (b) indicates the result obtained using K-Means. (c) shows the result obtained using iterated watersheds on the complete graph. (d) shows the result obtained using iterated watersheds with a constrained connectivity. (e) - (h) uses the make circles dataset [19]. (e) shows the original data/classes. (f) indicates the result obtained using K-Means. (g) shows the result obtained using iterated watersheds on the complete graph. (h) shows the result obtained using iterated watersheds with a constrained connectivity. **Note 1:** The results of iterated watersheds using complete graph and K-Means match. **Note 2:** When the connectivity is constrained, iterated watersheds obtained clusters closer to groundtruth.

In case of step (5), if the algorithm uses Floyd-Warshall to compute the center, then the time complexity increases exponentially. Using only the distances from boundary points, reduces the complexity depending on the data structure. In one extreme case of complete graph, this is equivalent to Floyd-Warshall, while for images/spatial data this would take only linear time.

4.2 Comparison with K-Means

Recall that the motivation of iterated watersheds was by taking the K-Means cost function and adding an additional constraint of connectivity to the problem. Thus, when considering a complete graph, one has that all vertices are adjacent to each other. Hence, in this case the results from K-Means and iterated watersheds algorithm should match. This is verified in Fig. 4.

When the connectivity is constrained, that is not all edges are considered, iterated watersheds is expected to perform on par with K-Means. This is reflected in comparison of the results in Fig. 4(f) and Fig. 4(h).

TABLE 1
Evaluation on Weizman datasets. The number of clusters is taken to be 2 for Weizman 1-Object dataset and 3 for Weizman 2-Object dataset. Gaussian similarity with parameter β varied over $\{1., 2., 3., 5.\}$ is used for Spectral Clustering and Isoperimetric partitioning. Only the best results among these parameter values are considered.

	Method	AMI	ARI	F	CA
1-Object	Iterated Watersheds	**0.2467**	**0.3126**	**0.7880**	0.8329
	Spectral Clustering	0.1674	0.1568	0.6889	**0.8697**
	Isoperimetric Partitioning	0.0712	0.0600	0.7772	0.7666
	K-Means	0.1811	0.2043	0.6684	0.8143
2-Object	Iterated Watersheds	**0.3675**	**0.3742**	0.7478	**0.8964**
	Spectral Clustering	0.2716	0.2636	**0.7576**	**0.8963**
	K-Means	0.2390	0.2099	0.5876	0.8873

4.3 Analysis using Image Segmentation

In this section, the proposed iterated watersheds is compared with other clustering methods on the problem of image segmentation. The results in this section are only used for better understanding of the performance and not to portray state-of-art results. In the next section, we shall see the main application of the proposed algorithm. The code to generate the results in this section and the next can be found in [20].

The problem of image segmentation is defined as - Given an image I, identify the subset of pixels assigned to the objects within the image. An edge weighted graph is constructed from the image as follows - each pixel is assigned a vertex, adjacent pixels are connected giving a 4-adjacency graph, and the weights are taken to be difference in the RGB values. The domain of image segmentation is useful for analysis of clustering methods, since several labelled datasets are available. Thus, image segmentation datasets - Weizman 1-Object and 2-Object datasets [21] are used for analysis.

To evaluate the iterated watersheds, the following methods are used for comparison - (i) Spectral clustering is considered since both iterated watersheds and spectral clustering can be interpreted as kernel K-Means as discussed in Section 2. (ii) Since, iterated watersheds are developed by adapting K-Means, simple K-Means is also used for comparison. (iii) Isoperimetric partitioning [22], a method related to spectral clustering is also considered. However, since isoperimetric partitioning partitions the image into two segments, only results on Weizman 1-Object results are considered.

Four evaluation metrics are considered - (i) *Adjusted Mutual Information (AMI)* [23] that calculates the mutual information adjusted for chance. (ii) *Adjusted Rand Index (ARI)* [24] that computes rand index adjusted to chance. (iii) *F-Score* (F_r in [25]) computes the harmonic mean of precision and recall. *Precision* denotes the ratio to the number of pairs of pixels, which have been predicted to have the same label indeed does in the groundtruth. *Recall* denotes the

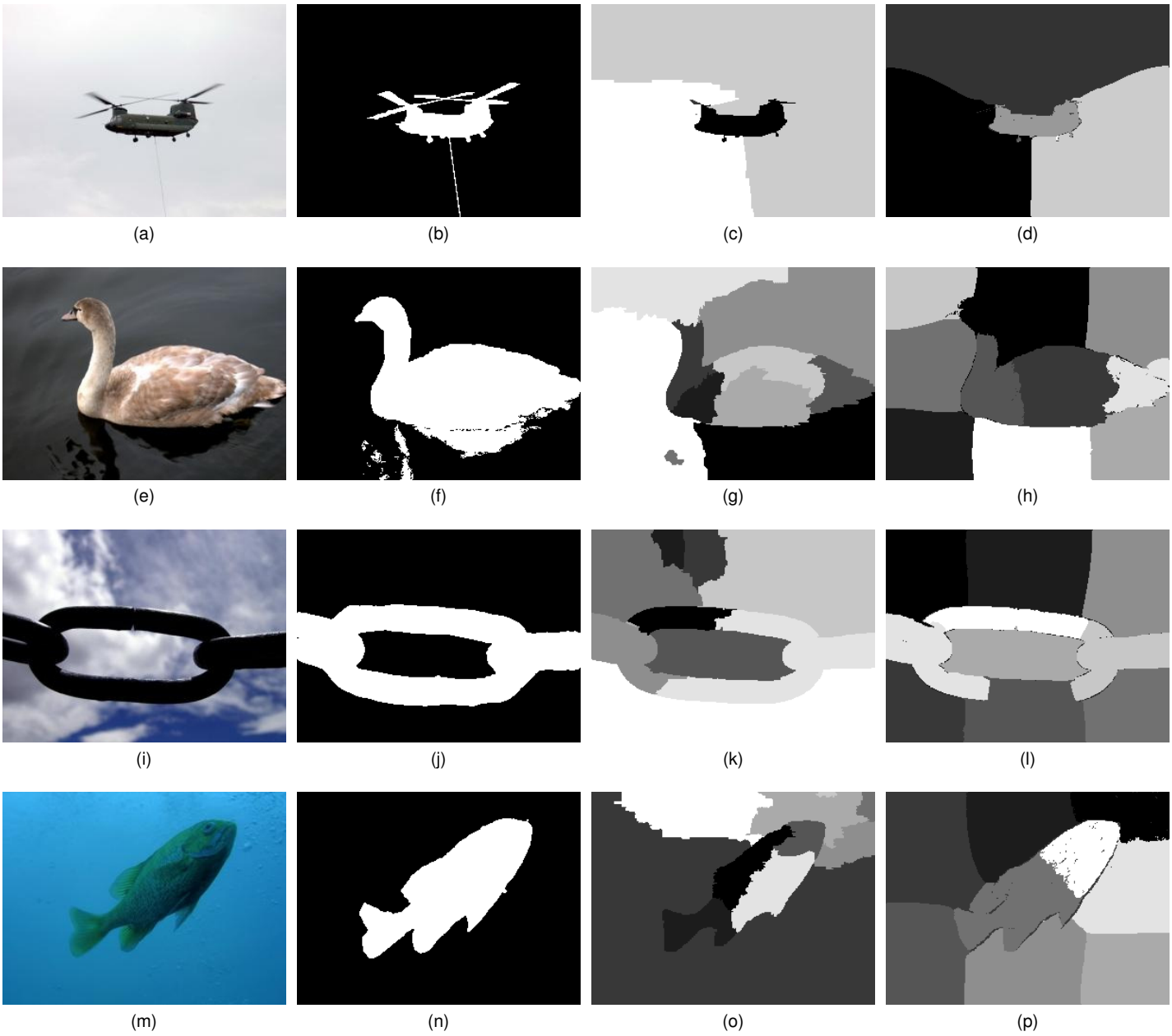
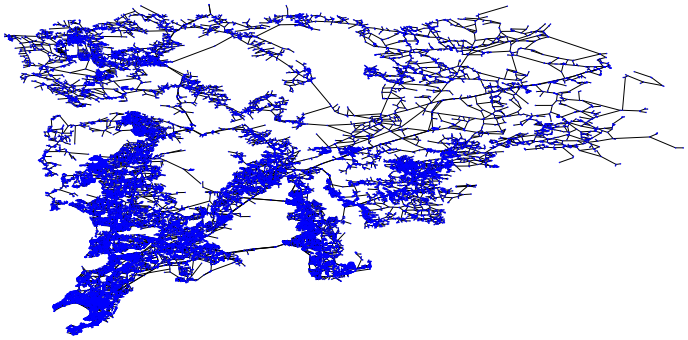
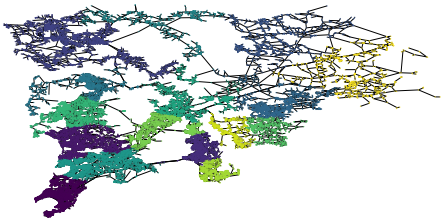


Fig. 5. Sample illustrative results on Weizman 1-Object Dataset. (a),(e),(i),(m) - Original Images. (b),(f),(j),(n) - Groundtruth images. (c),(g),(k),(o) - Segments obtained by iterative watersheds. (d),(h),(i),(p) - Segments obtained using spectral clustering.

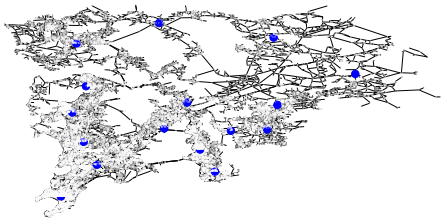
Fig. 6. Sample illustrative results on hyperspectral data. (a) Slice of the Pavia University hyperspectral image. (b) Predicted cluster of (a) using iterated watersheds. (c) Slice of the Salinas hyperspectral data. (d) Predicted clusters of (c) using iterated watersheds.



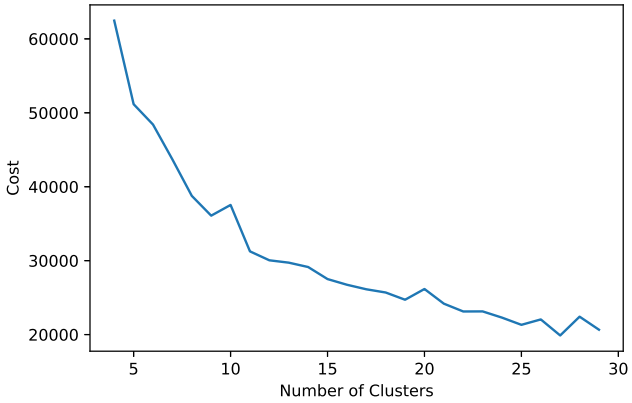
(a)



(b)



(c)



(d)

Fig. 7. Results on Mumbai Road Network Dataset. (a) Shows the complete road network in Mumbai. (b) Clustering obtained by the iterated watersheds algorithm considering the number of clusters as 16. (c) The centers of each of the clusters in (b) shown as blue dots. (d) The final cost after converging for varying number of clusters.

TABLE 2
Comparison of Iterated Watersheds with K-Means and Greedy K Centers on road network datasets.

City	Number Centers	Iterated Watersheds	K Means		K Centers	
			Cost	% Improvement	Cost	% Improvement
Mumbai	3	72065.43	78660.29	8.38	105022.57	31.38
	6	45424.89	64684.32	29.77	100682.75	54.88
	9	34447.86	52245.34	34.07	73536.33	53.16
	12	30329.42	42162.63	28.07	62578.70	51.53
	15	28107.24	32336.66	13.08	58229.78	51.73
Hyderabad	3	68570.63	109640.46	37.46	107685.94	36.32
	6	53022.46	62337.99	14.94	107112.82	50.50
	9	45730.64	47932.50	4.59	112123.11	59.21
	12	40298.87	47502.13	15.16	105458.64	61.79
	15	35929.86	42816.08	16.08	102670.83	65.00
Chennai	3	48618.69	144552.29	66.37	101894.35	52.29
	6	34292.08	39070.15	12.23	100403.69	65.85
	9	29105.73	34281.59	15.10	103995.35	72.01
	12	25520.35	30471.39	16.25	55257.34	53.82
	15	23736.54	26208.00	9.43	54228.99	56.23
Bengaluru	3	119816.38	219508.02	45.42	142934.18	16.17
	6	88490.71	100132.84	11.63	135937.67	34.90
	9	71256.02	77997.34	8.64	133889.24	46.78
	12	62615.27	74549.43	16.01	127551.37	50.91
	15	54755.50	63490.12	13.76	122551.30	55.32
Calcutta	3	21292.91	22532.27	5.50	40004.75	46.77
	6	15060.60	15752.89	4.39	39098.16	61.48
	9	12481.05	14292.80	12.68	37887.34	67.06
	12	11033.54	12306.89	10.35	37938.90	70.92
	15	10054.31	10709.78	6.12	32125.65	68.70
Delhi	3	58807.18	66272.17	11.26	100154.08	41.28
	6	41037.44	47832.93	14.21	101074.01	59.40
	9	33267.61	39463.66	15.70	64516.95	48.44
	12	28281.70	33253.71	14.95	58961.44	52.03
	15	25635.58	31785.57	19.35	51452.49	50.18

number of pairs of pixels, which have the same labels both in the groundtruth and in the prediction [25]. (iv) *Clustering Accuracy* [26] is computed by assigning each cluster, the class label with largest intersection with groundtruth. These results are shown in Table 1. Iterated watersheds perform as well as other methods across several measures, with the main gain shown in measures AMI and ARI. Compared to simple K-Means, iterated watersheds perform better since iterated watersheds consider the connectivity of the graph. Spectral clustering and isoperimetric partitioning penalize the smaller size of the clusters. This implies a tendency of these algorithms to combine pixels from adjacent regions. However, iterated watersheds do not penalize the size of the clusters. Images in Fig. 5(o) and Fig. 5(p) illustrate this effect. Also K-Means and spectral clustering do not ensure connectivity of the components (as can be seen Fig. 5(p)), while iterated watersheds ensure that the components are connected.

Recall that the time complexity of spectral clustering is approximately $\mathcal{O}(n^{3/2})$ [27]. However, the algorithm described here can be achieved in linear time. In summary, compared to spectral clustering, we have that iterated watersheds provides equivalent or better results with lesser time complexity.

For completeness, results on some slices on hyperspectral data are shown in Fig. 6. Observe that, while the algorithm can predict the high level structure, it could not identify small structures in the data.

5 Analysis of Road Network

As an application of the proposed algorithm, we consider the following problem - Given a road network, identify the 'ideal' points to establish emergency stations. The main responsibility of an emergency station is to reach the point of

incident as fast as possible. An ideal solution to the problem would have the following properties -

1) The distance from the point of incident to the emergency station must be as less as possible to allow for quick reaction time.
2) The number of emergency stations must be as less as possible to reduce the cost of establishment.

To obtain a solution, we start with constructing an edge weighted graph from the data, $G = (V, E, W)$. The set of vertices V indicates the reference points in the city under consideration. These reference points are used to calculate the distance, and response time. Also, we assume that a location of the emergency station would belong to this set. In this case, we have considered the vertices to be all the junction points in the graph. An alternate is to consider points within, say 1 kilometer, of each other. However, the approach remains the same.

The edge set E consists of tuples (x, y) such that x and y are connected by a road and there exists no other vertex between them. The edge weights W are taken to be distances between the end points. In case, information is available, one can consider the time taken to travel from one to another as well.

Using the construction as above, observe that the problem can be restated as finding a set of points $\{c_1, c_2, \cdots, c_k\}$, such that

$$\sum_{x \in V} d(x, M(x)) \tag{13}$$

where $M(x)$ denotes the nearest emergency station to x and $d(.,.)$ denotes the road distance. Observe that $M(x)$ belongs to $\{c_1, c_2, \cdots, c_k\}$. Also, the set $\{x \mid M(x) = c_i\}$ is connected for all i. Thus, this is equivalent to optimization problem in (10). And hence, iterated watersheds are applicable here. To identify the ideal number of stations, the number is increased and the final cost is observed. The ideal number of stations is at the point where the reduction in cost does not justify the cost of establishing another emergency station.

Here, we consider road networks of several cities taken from [6]. Figure 7(a) represents the road network of Mumbai. Figure 7(b) shows a clustering obtained, when the number of clusters is taken as 16. Figure 7(c) shows the corresponding centers for the best result obtained. The cost reduction with increase in number of emergency stations is plotted in Fig. 7(d).

To illustrate the performance of iterated watersheds, we consider two other approaches to solve this problem -

- **K-Means** is used to cluster the given dataset to obtain the centers - the proposed spots for emergency stations. Each of the other points is assigned to its nearest emergency station, and the cost is taken as a sum of all these distances.
- Note that the problem proposed here is equivalent to the K-center problem which is NP-Hard. **Greedy K-Center** is an approach which greedily picks the centers farthest from the already selected set of centers. See [28] for detailed analysis.

Table 2 shows the results obtained by various methods. The cost is computed using (13). Also, shown are the percentage improvement of iterated watersheds over other methods. From this, one can conclude that iterated watersheds perform better than baseline solutions. This can majorly be attributed to the fact that iterated watersheds can exploit the inherent graph structure compared to K-Means and Greedy K-Center methods. To corroborate this, observe that as the number of clusters are increasing, iterated watersheds reduce the cost uniformly. On the other hand, K-Means and Greedy K-Center methods do not lead to uniform reduction in the cost. However, as noted before, when a complete graph is considered there would not be any difference between iterated watersheds and K-Means.

6 Conclusion and Future Work

Several problems for efficient governance - zonation, river linking, facility location, visualization of GIS data, can be formulated as a clustering problem with an additional constraint that the clusters be connected. In this article, we propose a novel method to obtain these clusters, referred to as *Iterated Watersheds*. It is shown that iterated watersheds extend the classical K-Means by considering the additional constraint of connectivity which is common in several real life problems. This has been empirically verified as well by using results on complete graph, in which case iterated watersheds match with K-Means.

To analyze the proposed clustering algorithm, we consider the image segmentation problem since numerous labelled datasets are available. Iterated watershed is compared to spectral clustering, isoperimetric partitioning and K-Means on Weizman 1-Object and 2-Object datasets. It is shown that iterated watersheds outperform these techniques on a few measures, and fare comparably good on others.

However, as the main application area of the proposed algorithm is for clustering problems occurring in governance, we consider the road network dataset. A simple problem of placing emergency stations and suitable cost function is considered. Using real world road networks of various cities, iterated watersheds approach is compared with K-Means and greedy K-center methods. It is observed that iterated watersheds result in $4\% - 66\%$ improvement over K-Means and in $31\% - 72\%$ improvement over Greedy K-Centers in experiments on road networks of various cities. This illustrates that the proposed algorithm can efficiently cluster with constrained connectivity. This experiment also illustrates that the proposed algorithm can be widely used, thanks to its customizability.

In future we expect the current algorithm to be used on other datasets as well. For instance, one of the main problems of governance is understanding the root cause behind certain behavior. It is vital in such cases to visualize spatially distributed variables. Iterated watersheds can be used to cluster the space into significant 'zones' based on these variables and help in better visualization. Such applications constitute the future work. On the theoretical side, we hope to improve the algorithm for finding the center of the connected subgraph. The code is available at [20].

ACKNOWLEDGMENTS

Sampriti Soor, Aditya Challa and Sravan Danda would like to thank Indian Statistical Institute for support. The work of B. S. D. Sagar was supported by the Department of Science and Technology-Science and Engineering research Board (DST-SERB) and the Indian Space Research Organization (ISRO) under Grant EMR/2015/000853 SERB and Grant ISRO/SSPO/Ch-1/2016-17. L. Najman received funding from the Agence Nationale de la Recherche, contract ANR-14-CE27-0001 GRAPHSIP) and through "Programme d'Investissements d'Avenir" (LabEx BEZOUT ANR-10-LABX-58). We would like to thank the anonymous reviewers for their valuable comments, which have improved the quality of the article.

REFERENCES

[1] A. K. Jain, M. N. Murty, and P. J. Flynn, "Data clustering: A review," *ACM Comput. Surv.*, vol. 31, no. 3, pp. 264–323, 1999. [Online]. Available: https://doi.org/10.1145/331499.331504

[2] C. C. Aggarwal and C. K. Reddy, *Data Clustering: Algorithms and Applications*, 1st ed. Chapman & Hall/CRC, 2013.

[3] J. MacQueen *et al.*, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, vol. 1, no. 14. Oakland, CA, USA., 1967, pp. 281–297.

[4] D. Aloise, A. Deshpande, P. Hansen, and P. Popat, "Np-hardness of euclidean sum-of-squares clustering," *Machine Learning*, vol. 75, no. 2, pp. 245–248, 2009. [Online]. Available: https://doi.org/10.1007/s10994-009-5103-0

[5] S. Soor, A. Challa, S. Danda, B. S. D. Sagar, and L. Najman, "Extending k-means to preserve spatial connectivity," in *2018 IEEE International Geoscience and Remote Sensing Symposium, IGARSS 2018, Valencia, Spain, July 22-27, 2018*. IEEE, 2018, pp. 6959–6962. [Online]. Available: https://doi.org/10.1109/IGARSS.2018.8518643

[6] A. Karduni, A. Kermanshah, and S. Derrible, "A protocol to convert spatial polyline data to network formats and applications to world urban road networks," *Scientific Data*, vol. 3, pp. 160 046 EP –, 06 2016. [Online]. Available: http://dx.doi.org/10.1038/sdata.2016.46

[7] S. Lefèvre, "A new approach for unsupervised classification in image segmentation," in *Advances in Knowledge Discovery and Management [Best of EGC 2009, Strasbourg, France]*, ser. Studies in Computational Intelligence, F. Guillet, G. Ritschard, D. A. Zighed, and H. Briand, Eds., vol. 292. Springer, 2009, pp. 113–131. [Online]. Available: https://doi.org/10.1007/978-3-642-00580-0_7

[8] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the em algorithm," *Journal of the royal statistical society. Series B (methodological)*, pp. 1–38, 1977.

[9] U. von Luxburg, "A tutorial on spectral clustering," *Statistics and Computing*, vol. 17, no. 4, pp. 395–416, 2007. [Online]. Available: https://doi.org/10.1007/s11222-007-9033-z

[10] I. S. Dhillon, Y. Guan, and 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, Seattle, Washington, USA, August 22-25, 2004*, W. Kim, R. Kohavi, J. Gehrke, and W. DuMouchel, Eds. ACM, 2004, pp. 551–556. [Online]. Available: http://doi.acm.org/10.1145/1014052.1014118

[11] L. Najman and M. Schmitt, "Geodesic saliency of watershed contours and hierarchical segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 18, no. 12, pp. 1163–1173, 1996. [Online]. Available: https://doi.org/10.1109/34.546254

[12] J. Cousty, G. Bertrand, L. Najman, and M. Couprie, "Watershed cuts: Minimum spanning forests and the drop of water principle," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 8, pp. 1362–1374, 2009. [Online]. Available: https://doi.org/10.1109/TPAMI.2008.173

[13] ——, "Watershed cuts: Thinnings, shortest path forests, and topological watersheds," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 5, pp. 925–939, 2010. [Online]. Available: https://doi.org/10.1109/TPAMI.2009.71

[14] A. X. Falcão, J. Stolfi, and R. de Alencar Lotufo, "The image foresting transform: Theory, algorithms, and applications," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 1, pp. 19–29, 2004. [Online]. Available: http://doi.ieeecomputersociety.org/10.1109/TPAMI.2004.10012

[15] R. W. Floyd, "Algorithm 97: Shortest path," *Commun. ACM*, vol. 5, no. 6, pp. 345–, Jun. 1962. [Online]. Available: http://doi.acm.org/10.1145/367766.368168

[16] H. Djidjev, S. Thulasidasan, G. Chapuis, R. Andonov, and D. Lavenier, "Efficient multi-gpu computation of all-pairs shortest paths," in *2014 IEEE 28th International Parallel and Distributed Processing Symposium, Phoenix, AZ, USA, May 19-23, 2014*. IEEE Computer Society, 2014, pp. 360–369. [Online]. Available: https://doi.org/10.1109/IPDPS.2014.46

[17] J. B. Tenenbaum, V. d. Silva, and J. C. Langford, "A global geometric framework for nonlinear dimensionality reduction," *Science*, vol. 290, no. 5500, pp. 2319–2323, 2000. [Online]. Available: http://science.sciencemag.org/content/290/5500/2319

[18] "scikit-learn datasets," http://scikit-learn.org/stable/modules/generated/sklearn.datasets.make_blobs.html, accessed: 2017-12-12.

[19] "scikit-learn datasets," http://scikit-learn.org/stable/modules/generated/sklearn.datasets.make_circles.html, accessed: 2017-12-12.

[20] C. Aditya. Iterated watersheds. [Online]. Available: https://github.com/ac20/iteratedWatersheds

[21] S. Alpert, M. Galun, A. Brandt, and R. Basri, "Image segmentation by probabilistic bottom-up aggregation and cue integration," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 2, pp. 315–327, 2012. [Online]. Available: https://doi.org/10.1109/TPAMI.2011.130

[22] L. Grady and E. L. Schwartz, "Isoperimetric graph partitioning for image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 3, pp. 469–475, 2006. [Online]. Available: https://doi.org/10.1109/TPAMI.2006.57

[23] N. X. Vinh, J. Epps, and J. Bailey, "Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance," *J. Mach. Learn. Res.*, vol. 11, pp. 2837–2854, Dec. 2010. [Online]. Available: http://dl.acm.org/citation.cfm?id=1756006.1953024

[24] L. Hubert and P. Arabie, "Comparing partitions," *Journal of Classification*, vol. 2, no. 1, pp. 193–218, Dec 1985. [Online]. Available: https://doi.org/10.1007/BF01908075

[25] J. Pont-Tuset and F. Marqués, "Supervised evaluation of image segmentation and object proposal techniques," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 7, pp. 1465–1478, 2016. [Online]. Available: https://doi.org/10.1109/TPAMI.2015.2481406

[26] A. Fahad, N. Alshatri, Z. Tari, A. Alamri, I. Khalil, A. Y. Zomaya, S. Foufou, and A. Bouras, "A survey of clustering algorithms for big data: Taxonomy and empirical analysis," *IEEE Trans. Emerging Topics Comput.*, vol. 2, no. 3, pp. 267–279, 2014. [Online]. Available: https://doi.org/10.1109/TETC.2014.2330519

[27] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 8, pp. 888–905, 2000. [Online]. Available: https://doi.org/10.1109/34.868688

[28] V. V. Vazirani, *Approximation algorithms*. Springer Science & Business Media, 2013.

Sampriti Soor is currently working as a Senior Research Fellow at Systems Science and Informatics Unit, Indian Statistical Institute. He received his M.E. degree from Jadavpur University, Kolkata, in Software Engineering in 2015. His research interests are in the fields of Data Science, Image Processing and Mathematical Morphology.

Aditya Challa received the B.Math.(Hons.) degree in Mathematics from the Indian Statistical Institute - Bangalore, and Masters in Complex Systems from University of Warwick, UK - in 2010, and 2012, respectively. From 2012 to 2014, he worked as a Business Analyst at Tata Consultancy Services, Bangalore. In 2014, he joined as a Junior Research Fellow at Indian Statistical Institute - Bangalore, where he is currently a Senior Research Fellow in the Systems Science and Informatics Unit. His current research interests focus on the solutions to the Image Interpolation Problem, and using techniques from Mathematical Morphology in Data Mining.

Sravan Danda received the B.Math.(Hons.) degree in Mathematics from the Indian Statistical Institute - Bangalore, and the M.Stat. degree in Mathematical Statistics from the Indian Statistical Institute - Kolkata, in 2009, and 2011, respectively. From 2011 to 2013, he worked as a Business Analyst at Genpact - Retail Analytics, Bangalore. In 2013, he joined as a Junior Research Fellow at Indian Statistical Institute - Bangalore, where he is currently a Senior Research Fellow in the Systems Science and Informatics Unit under the joint supervision of B.S.Daya Sagar and Laurent Najman. His current research interests are discrete mathematical morphology and discrete optimization.

B. S. Daya Sagar (M'03-SM'03) is a full Professor of the Systems Science and Informatics Unit (SSIU) at the Indian Statistical Institute. Sagar received the M.Sc and Ph.D degrees from the Faculty of Engineering, Andhra University, Visakhapatnam, India, in 1991 and 1994 respectively. He is also the first Head of the SSIU. Earlier, he worked in College of Engineering, Andhra University, and Centre for Remote Imaging Sensing and Processing (CRISP), The National University of Singapore in various positions during 1992-2001. He served as Associate Professor and Researcher in the Faculty of Engineering and Technology (FET), Multimedia University, Malaysia during 2001-07. His research interests include mathematical morphology, GISci, digital image processing, fractals and multifractals their applications in extraction, analyses, and modeling of geophysical patterns. He has published over 85 papers in journals, and has authored and/or guest edited 11 books and/or special theme issues for journals. He recently authored a book entitled "Mathematical Morphology in Geomorphology and GISci," CRC Press: Boca Raton, 2013, p. 546. He recently co-edited a special issue on "Filtering and Segmentation with Mathematical Morphology" for IEEE Journal on Selected Topics in Signal Processing (v. 6, no. 7, p. 737-886, 2012). His recent book on "Handbook of Mathematical geosciences: Fifty Years of IAMG", Springer Publishers, p. 942, 2018 crossed the downloads of 90000 within three months of its release. He is an elected Fellow of Royal Geographical Society (1999), Indian Geophysical Union (2011), and was a member of New York Academy of Science during 1995-96. He received Dr. Balakrishna Memorial Award from Andhra Pradesh Akademi of Sciences in 1995, Krishnan Gold Medal from Indian Geophysical Union in 2002, and 'Georges Matheron Award-2011 (with Lecturership)" of International Association for Mathematical Geosciences (IAMG), and IAMG Certificate of Appreciation - 2018. He is the Founding Chairman of Bangalore Section IEEE GRSS Chapter. He is on the Editorial Boards of Computers and Geosciences, and Frontiers: Environmental Informatics. More details about him can be seen at http://www.isibang.ac.in/ bsdsagar

Laurent Najman (SM'17) received the Habilitation à Diriger les Recherches in 2006 from University of Marne-la-Vallée, a Ph.D. of applied mathematics from Paris-Dauphine University in 1994 with the highest honor (Félicitations du Jury) and an "Ingénieur" degree from the Ecole des Mines de Paris in 1991. After earning his engineering degree, he worked in the central research laboratories of Thomson-CSF for three years, working on some problems of infrared image segmentation using mathematical morphology. He then joined a start-up company named Animation Science in 1995, as director of research and development. The technology of particle systems for computer graphics and scientific visualization, developed by the company under his technical leadership received several awards, including the "European Information Technology Prize 1997" awarded by the European Commission (Esprit programme) and by the European Council for Applied Science and Engineering and the "Hottest Products of the Year 1996" awarded by the Computer Graphics World journal. In 1998, he joined OCÉ Print Logic Technologies, as senior scientist. He worked there on various problem of image analysis dedicated to scanning and printing. In 2002, he joined the Informatics Department of ESIEE Paris, where he is professor and a member of the Laboratoire d'Informatique Gaspard Monge, Université Paris-Est Marne-la-Vallée. His current research interest is discrete mathematical morphology and discrete optimization.