Power Spectral Clustering

Aditya S Challa, Sravan Danda, B S S Daya Sagar, Laurent Najman

Power Spectral Clustering

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

Abstract—Clustering is an important part of several data mining tasks. Due to its unsupervised nature it has several applications ranging from document classification to image segmentation. Recent trend towards large datasets have required faster and scalable algorithms for clustering tasks. One solution to the clustering problem is offered by the method of spectral clustering, which has received wide attention due to its ability to detect non convex clusters. In this article, we propose a faster alternative to the spectral clustering method, which is obtained by considering the $\Gamma-$limit of spectral clustering methods. The proposed method, referred to as PRcut, is analyzed and compared with spectral clustering and MST based clustering methods. We illustrate with experiments that PRcut is superior to its counterpart in spectral clustering in terms of speed and adaptability.

Index Terms—Clustering, Spectral Clustering, $\Gamma-$convergence, MST based clustering, Multi scale combinatorial grouping.

1 Introduction

The problem of clustering can be stated as - given a data set $\{x_i\}$, partition the set into groups such that elements in the same group are 'similar' and elements in different sets are 'dissimilar'. This has been a problem of interest since the early 20^{th} century and is widely applicable in several domains such as text categorization [1], anomaly detection [2], market research [3] and image segmentation [4], [5] etc. The vast literature available on the topic of clustering shows the extent of importance in real-world applications. A detailed analysis of various aspects of clustering and its history can be found in [4], [6], [7]. A solution to the clustering problem is given by the use of spectral methods, referred to as *spectral clustering* [8]. Contrasting with the popular k-means algorithm [7], spectral clustering methods have the ability to detect non-convex clusters. These methods proceed by constructing an edge-weighted graph from the data and use the eigenvectors of the Laplacian of the graph. However, the calculation of the eigenvectors is computationally expensive and is prone to errors. Several efforts were made to increase the speed and accuracy of the spectral clustering methods [9], [10], [11], [12], including parallelizing [13]. In [14], the authors propose to reduce the size of the similarity matrix to speed up the normalized cuts procedure for image segmentation [5].

Another clustering method which allows to detect non-convex clusters is that of *Maximum Spanning Tree (MST) based clustering* [15]. These methods proceed by constructing an MST on the edge weighted graph and removing the edges with least similarity. The main advantage of this method is that it is fast and scales well for big data. However, these methods also have some ambiguity in clusters and results in degenerate solutions. (MST based clustering is discussed in detail in section 2.)

In this article, the aim is to develop an algorithm which uses efficient MST based clustering within a cluster and more computationally expensive spectral clustering near the borders of the clusters. This is achieved by limit of minimizers (a.k.a $\Gamma-$limit) of spectral clustering to obtain the faster version of spectral clustering while preserving the properties of spectral clustering. This is referred to as *power spectral clustering* or *Power Ratio cut* or more simply *PRcut*. The study of $\Gamma-$limits is referred to as the $\Gamma-$convergence. $\Gamma-$convergence is widely used in the areas of computer vision and in the field of calculus of variations (See chapter 5 of [16]). In [17], the authors proposed *Power Watershed* - the $\Gamma-$limit of the energy function in [18] and demonstrated its similarity with watershed transformation [19], [20]. It turns out the Power Watershed performs better in several cases compared to the usual seeded segmentations. In [21], the power watershed framework was extended and a simple generic algorithm was proposed to calculate the limit of minimizers under some conditions. In this article, we propose to use the extended power watershed framework to calculate the $\Gamma-$limit of spectral clustering.

Our main contributions are-

1) We provide an efficient implementable algorithm to calculate the $\Gamma-$limit of the spectral clustering, PRcut.
2) PRcut is compared with two closest clustering methods - MST based clustering and Spectral Clustering. It is shown that PRcut outperforms MST based clustering. Compared to spectral clustering, PRcut is shown to preserve the properties of the spectral clustering while being a faster alternative.
3) PRcut is used in the method multiscale combinatorial grouping [14] in place of the normalized cuts and shown to outperform normalized cuts with respect to speed while preserving accuracy.
4) Clustering algorithms are extremely domain dependent [4]. Hence, versatility would be an added advantage of a clustering algorithm. We show that PRcut algorithm is versatile enough to be adapted without compromising performance by applying it on hyperspectral datasets.

The outline of the article is as follows - In section 2, we

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: aditya.challa.20@gmail.com, sravan8809@gmail.com, bsdsagar@isibang.ac.in
Laurent Najman is with Université Paris-Est, LIGM, Equipe A3SI, ESIEE, France. E-mail: laurent.najman@esiee.fr

introduce various concepts required for the rest of the article - spectral clustering, MST clustering and $\Gamma-$convergence. In section 3, we start with a generic algorithm to calculate the $\Gamma-$limit and characterize different parts of the algorithm to obtain an implementable version. In section 4 we further increase the efficiency by identifying the commonality between eigenvectors and connected components of the Laplacian. In section 5, we explore the PRcut in detail. We compare PRcut with MST based clustering and spectral clustering. PRcut is used in the context of multiscale combinatorial grouping (MCG) [14] in place of normalized cuts, and time and accuracy analysis is considered. Applications of PRcut for hyperspectral data and high dimensional data is also demonstrated.

2 Background

Let $X = \{x_i\}$ represents the dataset, where x_i denotes each data entry. Let $x_i \in \mathbb{R}^f$, where f is the number of features. One can construct a similarity graph $\mathcal{G} = (V, E, W)$. The vertex set, V, is taken to be the set of all data points, each point x_i is represented by a vertex. The edge set E is a subset of $V \times V$. $W : E \to \mathbb{R}^+$ denotes weights (similarities) assigned to each edge, where $\mathbb{R}^+$ denotes the set of positive real numbers. w_{ij} denotes the similarity between x_i and x_j. There are several methods to construct the graph from the dataset depending on the domain of application [8].

D is a diagonal matrix, $diag(d_1, d_2, \cdots, d_n)$, such that

$$d_i = \sum_j w_{ij} \tag{1}$$

The *Laplacian* of a graph is defined by

$$L = D - W \tag{2}$$

We know that the Laplacian is a symmetric positive-semi definite matrix, and hence has non-negative real eigenvalues. The eigenvalues are represented by $0 = \lambda_0 \leq \lambda_1 \leq \cdots \leq \lambda_{n-1}$. The corresponding eigenvectors are denoted by $\{e_0, e_1, \cdots, e_{n-1}\}$. Let $A \subseteq V$. Denote

$$\mathbf{1}_A(x) = \begin{cases} 1 & \text{if } x \in A \\ 0 & \text{otherwise} \end{cases}$$

$\mathcal{G}_{\geq w}$ denotes the *thresholded* graph, with vertex set V and edge set, $E_{\geq w}$, consisting of only those edges whose weight is greater than or equal to w.

2.1 Spectral Clustering

Spectral clustering methods work by projecting the data onto a subspace, so that similar points are close by and dissimilar points are far apart in the projected subspace. There are 3 steps which form the core of spectral clustering methods-

1) Given a set of points $\{x_i\}$ (dataset), construct a graph, $\mathcal{G}$, with each data point as a vertex.
2) Construct the Laplacian for the obtained graph and calculate the first k eigenvectors of the Laplacian. The value of k is fixed based on the number of clusters required. Let H be the matrix such that the i^{th} column of H is the i^{th} eigenvector e_{i-1}.
3) Using rows of the matrix H as new representation of the points x_i, use classical clustering methods such as k-means to obtain the final clusters.

The Laplacian in (2) is known as an unnormalized Laplacian. Some works also consider the normalized Laplacians, L_1, L_2 as well [5], [22], where,

$$L_1 = I - D^{-1}W \quad L_2 = I - D^{-1/2}WD^{-1/2} \tag{3}$$

Why spectral clustering works? The idea behind working of spectral clustering methods can be intuitively understood using two results - (i) Suppose the graph $\mathcal{G}$ has k connected components, $\{A_1, A_2, \cdots, A_k\}$. Then the basis of the vector space spanned by the first k eigenvectors is $\{\mathbf{1}_{A_1}, \mathbf{1}_{A_2}, \cdots, \mathbf{1}_{A_k}\}$ and (ii) The eigenspaces of a matrix and its perturbation are 'close' [8]. Assume that the similarity between points from two different clusters is small and similarity between points from same cluster is high. Then, from the above results, the first few eigenvectors of the Laplacian are close to the indicator of the clusters. Hence simple clustering methods work well on this projected data.

Different formulations of spectral clustering

The spectral clustering can also be analyzed in an optimization framework. Given a similarity graph $\mathcal{G} = (V, E, W)$, consider the *graph cut* measure

$$cut(A_1, A_2, \cdots, A_k) = \frac{1}{2}\sum_{i=1}^{k} W(A_i, \overline{A_i}) \tag{4}$$

where

$$W(A, B) = \sum_{i \in A; j \in B} w_{ij} \tag{5}$$

$\overline{A}$ denotes the complement of A in the vertex set V. k denotes the size of the partition required. Note that $cut(.,.)$ measures how similar the clusters are by taking the sum of the weights of the edges connecting distinct clusters, and hence can be used to partition a graph. In practice, however, minimizing the $cut(.,.)$ does not give good results, since it generally separates one vertex, and gives degenerate solutions. Also, minimizing $cut(.,.)$ for $k \geq 3$ is NP-hard [23]. Thus, it was proposed to use a variation of the above cost function. *Ratio cut* [8] is given by

$$Rcut(A_1, A_2, \cdots, A_k) = \frac{1}{2}\sum_{i=1}^{k} \frac{W(A_i, \overline{A_i})}{|A_i|} \tag{6}$$

known as *Ratio cut*. Here $|A_i|$ is the cardinality of set A_i. Note that ratio cut penalizes small clusters and hence avoids degenerate solutions. Let $\mathcal{G}$ be a graph constructed from the data set, and let L denote its corresponding Laplacian. It is shown that minimizing the $Rcut(.,.)$ for k clusters is approximately equivalent to solving the optimization problem in (7) [8].

$$\begin{aligned} \underset{H \in \mathbb{R}^{n \times k}}{\text{minimize}} \quad & Tr(H^t L H) \\ \text{subject to} \quad & H^t H = I \end{aligned} \tag{7}$$

where, I is the identity matrix. From the *Rayleigh-Ritz* theorem it is known that the solution to optimization problem in (7) is obtained by considering the first k eigenvectors of L as columns of H [24].

Another variation of the $cut(.,.)$ cost function, proposed in [5] is the *Normalized cut* (Ncut) cost function given by

$$Ncut(A_1, A_2, \cdots, A_k) = \frac{1}{2}\sum_{i=1}^{k}\frac{W(A_i, \overline{A_i})}{vol(A_i)} \tag{8}$$

where $vol(A_i) = \sum_i d_i$. It can be shown that for k clusters, $Ncut(.,.)$ is approximately equivalent to [8]

$$\begin{aligned} \underset{H}{\text{minimize}} \quad & Tr(H^t L H) \\ \text{subject to} \quad & H^t D H = 1 \end{aligned} \tag{9}$$

It is known that the solution to this optimization problem is obtained by taking the first k eigenvectors of $I - D^{-1}W$ as columns of H [24].

2.2 Maximum Spanning Tree Clustering

Another method of clustering which can detect non globular clusters in the data is *maximum spanning tree based clustering*. Let $\mathcal{G} = (V, E, W)$ be a connected similarity graph. A *spanning tree*, T, is a connected acyclic subgraph of $\mathcal{G}$ whose vertex set is V. Each spanning tree can be assigned a numerical value by taking the sum of the weights of the edges in the tree,

$$w(T) = \sum_{e \in E(T)} w(e) \tag{10}$$

A spanning tree with the maximum weight is known as the maximum spanning tree (MST). There are several clustering methods based on MST [15]. Although MST based clustering is not so prevalent in clustering general data, in the context of image segmentation, slight variations of MST based clustering was shown to be very useful, thanks to its equivalence to watersheds [19], [20]. A generic MST based clustering algorithm is

1. Given a similarity graph $\mathcal{G} = (V, E, W)$, construct a MST, T, and sort the edges of T according to the weights.
2. If the required number of clusters is equal to m, add edges starting from highest weight until the number of components m are reached. The ties are broken arbitrarily.

MST based clustering is categorized under graph based clustering and is related to the hierarchical approaches. In fact, MST based clustering as described above is equivalent to single link hierarchical clustering.

The main problems with MST based clustering are - (i) It is prone to noise and outliers (ii) In practice, it gives small clusters or degenerate solutions which are not so meaningful (iii) Arbitrariness in breaking of ties results in non uniform solutions. In the later sections it will be seen that the proposed method, PRcut, is similar to MST based clustering and does not suffer from these problems.

2.3 Γ-convergence

Let $\min\{F_m(x) : x \in X\}$ be a family of minimum problems. A question of interest is the limiting behavior of the minimizers of this family as $m \to \infty$. Ideally this would be substituted by a single minimum problem $\min\{F(x) : x \in X\}$ which captures the limiting behavior. Few of the advantages of such a substitution are - 1) This gives an approximate solution to the family of minimizers which are much harder to calculate than the limit 2) Dependence on a parameter is nullified 3) This results in a new method which would present a different model which was previously modeled with F_m. The theory of $\Gamma-$convergence focusses on understanding the conditions under which such a substitution is possible [16].

In this article, however, we are interested in calculating a $\Gamma-$limit of the spectral clustering discussed above. Recently the authors in [17] calculated $\Gamma-$limit of the seeded random walker cost function [18], and proposed *Power watershed* seeded segmentation. This $\Gamma-$limit involves an MST, and hence links the random walk segmentation with MST clustering. In [21], a generic algorithm was proposed to calculate a $\Gamma-$limit, which we review here.

Let $0 < \lambda_1 < \lambda_2 < \cdots < \lambda_k \leq 1$ and $Q_i(x)$ is continuous for all i. Consider the cost function

$$Q^p(x) = \sum_{i=0}^{k} \lambda_i^p Q_i(x) \tag{11}$$

The problem is to calculate the limit of minimizers of $Q^{(p)}(x)$ as $p \to \infty$. Observe that the function itself converges to 0 at every point if $\lambda_k < 1$, and hence minimizers of the limit would be the whole space. Let C be a compact set. For sake of simplicity, assume that we are interested in finding the solutions in C. Define

$$M_k = \arg\min_{x \in C} Q_k(x) \tag{12}$$

i.e, M_k is the set of minimizers for Q_k. Now recursively define,

$$M_i = \arg\min_{x \in M_{i+1}} Q_i(x) \tag{13}$$

Theorem 1 ([21]). *Let X^* be the union of the sets of minimizers of $Q^{(p)}$ (as defined in* (11)*) for all p. Then every limit point of X^* belongs to the set M_1.*

The above theorem can be interpreted in terms of *scale*. If one interprets each λ_i as a scale, then theorem 1 states - the $\Gamma-$limit (limit of minimizers) belongs to the set of solutions which minimizes all the cost functions at different scales λ_i starting from the largest λ_i. The main consequence of theorem 1 is that one can now have algorithm 1 to calculate a $\Gamma-$limit [21].

Algorithm 1 Generic Algorithm to Compute $\Gamma-$limit.

Input: Function $Q^{(p)}(x) = \sum_{i=1}^{k} \lambda_i^p Q_i(x)$, where $0 \leq \lambda_1 < \lambda_2 < \cdots < \lambda_k \leq 1$.
Output: M_1
1: $M_k = \arg\min Q_k(x)$ where $x \in C$
2: **for** i from k to 1 **do**
3: Compute $M_i = \arg\min Q_i(x)$ where $x \in M_{i+1}$
4: **end for**

Theorem 1 ensures that a $\Gamma-$limit belongs to the output of algorithm 1. The converse is not true in general, i.e, all solutions obtained from algorithm 1 need not be a $\Gamma-$limit. However, it can be shown that they are equivalent. (**Remark:** 'equivalent' in the sense that the value of the cost function is same for all p, and hence in the limit as well.)

Proposition 1. *Let x^* be a $\Gamma-$limit for Q^p. Let $\hat{x}$ be a solution obtained from algorithm 1. Then we have that, for all p*

$$Q^p(x^*) = Q^p(\hat{x}) \tag{14}$$

The proof of the proposition 1 is straightforward. Although algorithm 1 might not calculate a $\Gamma-$limit, we have that it calculates equivalent solution as far as the cost function is concerned.

3 $\Gamma-$LIMIT OF SPECTRAL CLUSTERING

In several cases discrete models are used for modeling the data, while the data might be continous. This is especially true in the case of graph-based models [17]. Thus a *discretization scheme* is required to be able to use to discrete models for real data. We introduce the definition of a discretization scheme which is then used for calculating the $\Gamma-$limit.

Let $G = (V, E, \hat{W})$ be a graph. Recall that $\hat{W} : E \to \mathbb{R}^+$.

Definition 1 (Discretization Scheme)**.** *A discretization scheme is a decomposition of $\hat{W}$,*

$$\hat{W} = W \times \omega \tag{15}$$

where $W : E \to \mathbb{Z}^+$ and $\omega : E \to \mathbb{R}^+$, with a condition that if $\hat{W}(e_1) < \hat{W}(e_2)$, then $W(e_1) < W(e_2)$.

An example of a discretization scheme is given by taking the least integer function.

$$\hat{W}(e) = \lfloor \hat{W}(e) \rfloor \times \frac{\hat{W}(e)}{\lfloor \hat{W}(e) \rfloor} = W(e) \times \omega(e) \tag{16}$$

Here $W(e) = \lfloor \hat{W}(e) \rfloor$ and $\omega(e) = \hat{W}(e)/\lfloor \hat{W}(e) \rfloor$. Intuitively, W is the discrete version of $\hat{W}$ and ω is error with respect to the discretization. It is also clear that there are several discretization schemes possible.

An *exponentiated graph* is denoted by $\mathcal{G}^{(p)} = (V, E, \hat{W}^{(p)})$, where $\hat{W}^{(p)}(e) = (W(e))^p \omega(e)$. Accordingly we can define $D^{(p)}$, a diagonal matrix,

$$[D^{(p)}]_{ii} = \sum_j \hat{W}^{(p)}_{ij} \tag{17}$$

and a Laplacian $L^{(p)} = D^{(p)} - \hat{W}^{(p)}$.

We define the $\Gamma-$limit of spectral clustering by the limit of minimizers of the optimization problem (18) as $p \to \infty$.

$$\begin{array}{ll} \underset{H \in \mathbb{R}^{n \times m}}{\text{minimize}} & Tr(H^t L^{(p)} H) \\ \text{subject to} & H^t H = I \end{array} \tag{18}$$

Since the datasets are finite, W takes only finite number of values. Let these values be denoted by $0 < w_1 < w_2 < \cdots < w_k < 1$ indicating the k distinct weights $W(e)$ can take. Denote by $\mathcal{G}_i$ the graph with the vertex set, V, and the edge set consisting of only those edges e such that $W(e) = w_i$. The weight of an edge e in $\mathcal{G}_i$ is given by the function $\omega(e)$. The Laplacian for the graph $\mathcal{G}_i$ is denoted by L_i. Then it is easy to see that

$$L = \sum_{i=1}^{k} w_i L_i \tag{19}$$

and hence,

$$Tr(H^t L H) = \sum_{i=1}^{k} w_i Tr(H^t L_i H) \tag{20}$$

Accordingly, for the exponentiated graphs we have

$$Tr(H^t L^{(p)} H) = \sum_{i=1}^{k} w_i^p Tr(H^t L_i H) \tag{21}$$

Note that the above equation is in the form of (11) and hence we can use the generic algorithm 1 to calculate the $\Gamma-$limit.

Let $\overline{M}$ denote the set of all possible m dimensional subspaces of $\mathbb{R}^n$. Each subspace in $\overline{M}$ can be associated with a $n \times m$ matrix H, such that the column space of H is the subspace. Note that this matrix is not unique. Also, let $\mathcal{H}(M)$ denote the set of all matrices whose column space is equivalent to any of the subspaces in the set M. Let $\mathcal{M}(H)$ denote the set of all column spaces of matrices in a set of matrices H. It is easy to see that the following relations hold true.

$$\begin{array}{rcl} \mathcal{M}(\mathcal{H}(M')) & = & M' \\ \mathcal{H}(\mathcal{M}(H')) & \supseteq & H' \end{array}$$

Algorithm 2 Generic Algorithm to Compute $\Gamma-$limit.

Input: A weighted graph, $\mathcal{G}$, with distinct weights $w_1 < w_2 < \cdots < w_k$. Number of clusters to calculate m.
Output: M_1

1: Set $i = k$, $M_i = \overline{M}$
2: Construct the graph $\mathcal{G}_i$ at level i, and Laplacian L_i.
3: Solve the optimization problem

$$\begin{array}{ll} \underset{H \in \mathbb{R}^{n \times m}}{\text{minimize}} & Tr(H^t L_i H) \\ \text{subject to} & H^t H = \mathbf{I} \\ & H \in \mathcal{H}(M_i) \end{array} \tag{22}$$

4: Let H_i be the set of possible minimizers of the above optimization problem. Set $M_{i-1} = \mathcal{M}(H_i)$.
5: Set i = i - 1
6: **if** i = 0 **then**
7: Stop.
8: **return** M_1
9: **else**
10: Goto Step (2)
11: **end if**

where M' is a set of subspaces and H' is a set of matrices. With this notation, the generic algorithm 1 in the case of ratio cut optimization would be the one in algorithm 2.

However, this algorithm is not implementable. One needs to characterize all the solutions to the optimization problem in (7) to obtain an implementable version. This will in turn characterize the M_i and $\mathcal{H}(M_i)$ at every stage i of the algorithm 2.

Solutions to ratio cut optimization problem

Given a Laplacian L of dimensions $n \times n$, let $\lambda_1 < \lambda_2 < \cdots \lambda_n$ denote the eigenvalues and $\{e_1, e_2, \cdots, e_n\}$ denote the corresponding eigenvectors. Let $\lambda_{(m)}$ denote the m^{th}

smallest eigenvalue. Let A be the matrix obtained by stacking all the eigenvectors whose eigenvalue is less than or equal to $\lambda_{(m)}$. Assuming that there are l such eigenvectors, the dimension of A would be $n \times l$. Here m is the number of clusters required. Let l_2 be the number of eigenvectors whose eigenvalue is exactly equal to $\lambda_{(m)}$, and l_1 be the number of remaining vectors. We then have $l_1 + l_2 = l$. Let K be the matrix

$$K = \begin{bmatrix} I_{l_1 \times l_1} & 0 \\ 0 & X_{l_2 \times m - l_1} \end{bmatrix} \tag{23}$$

where $K^t K = I$. Let Y be any orthogonal matrix. Theorem 2 characterizes the sets $\mathcal{H}(M_i)$ and M_i at every stage.

Theorem 2. *The set of all solutions to the optimization problem* (7) *is the set of all solutions of the form* AKY.

Now, starting at highest level k (edge set consisting only of edges with weight w_k) we have $M_k = \overline{M}$. From theorem 2 we know that all the solutions to the optimization problem (22) for $i = k$ is of the form $A_k KY$, where A_k is the matrix of eigenvectors for L_k constructed as before. At level $k-1$, we have $M_{k-1} = \mathcal{M}(A_k KY)$. Now, we have the following lemma.

Lemma 1. *Given the notation as above, we have*

$$\mathcal{H}(\mathcal{M}(A_k KY)) = \{ \textit{ matrices of the form } A_k KY \} \tag{24}$$

Thus at level $k-1$ we need to solve the optimization problem,

$$\begin{aligned} \underset{H \in \mathbb{R}^{n \times m}}{\text{minimize}} \quad & Tr(H^t L_{k-1} H) \\ \text{subject to} \quad & H^t H = \mathbf{I} \\ & H \sim A_k KY \end{aligned}$$

(**Remark:** $H \sim A_k KY$ is read H is of the form $A_k KY$) which is equivalent to solving the optimization problem

$$\begin{aligned} \underset{K}{\text{minimize}} \quad & Tr(Y^t K^t A_k^t L_{k-1} A_k KY) \\ \text{subject to} \quad & K^t K = \mathbf{I} \end{aligned}$$

Now, since Y is orthogonal, we have

$$Tr(Y^t K^t A_k^t L_{k-1} A_k KY) = Tr(K^t A_k^t L_{k-1} A_k K)$$

Noting the special form of K as in (23), solving the above optimization problem is equivalent to solving the optimization problem

$$\begin{aligned} \underset{X}{\text{minimize}} \quad & Tr(X^t [A_k^t L_{k-1} A_k]_{l_2 \times l_2} X) \\ \text{subject to} \quad & X^t X = \mathbf{I} \end{aligned}$$

where, $[A_k^t L_{k-1} A_k]_{l_2 \times l_2}$ is the lower-right $l_2 \times l_2$ block. Here l_2 indicates the number of times the eigenvalue $\lambda_{(m)}$ repeats in L_k. Also matrix $[A_k^t L_{k-1} A_k]_{l_2 \times l_2}$ is symmetric positive semi-definite. Hence theorem 2 applies. Call $\hat{A}_{k-1}$ the matrix obtained by stacking the eigenvectors of $[A_k^t L_{k-1} A_k]_{l_2 \times l_2}$. Then all the solutions are of the form $\hat{A}_{k-1} KY$. Let

$$A_{k-1} = \begin{bmatrix} I_{l_1 \times l_1} & 0 \\ 0 & \hat{A}_{k-1} \end{bmatrix} \tag{25}$$

Then we have that H_{k-1} is the set of all matrices which are of the form $A_k A_{k-1} KY$. This follows from the fact that the matrices K and Y are arbitrary under the constraint that $K^t K = I$ and Y is some orthogonal matrix. Repeating this procedure, one can can obtain an algorithm to calculate the $\Gamma-$limit. This is summarized in algorithm 3.

Algorithm 3 Algorithm to Compute $\Gamma-$limit for Ratio-cut.

Input: A weighted (similarity) graph, $\mathcal{G}$, with distinct weights $w_1 < w_2 < \cdots < w_k$. Number of clusters, m.
Output: N_1

1: Set $i = k$, $N_i = \mathbf{I}_n$, $l_1 = 0$, $l_2 = n$ {l_2 indicates the number of eigenvectors at the end whose eigenvalue is the same.}
2: Construct the graph $\mathcal{G}_i$ at level i, and Laplacian L_i.
3: Construct matrix $C = [N_i^t L_i N_i]_{l_2, l_2}$
4: Calculate the first eigenvectors of the generalized eigenvalue problem

$$Cx = \lambda x \tag{26}$$

such that the eigenvalue is less than or equal to $\lambda_{(m-l_1)}$. Let A be the matrix with the eigenvectors stacked as columns.
5: Construct $\hat{A}$ as

$$\hat{A} = \begin{bmatrix} \mathbf{I} & 0 \\ 0 & A \end{bmatrix}$$

6: Set $N_{i-1} = N_i \hat{A}$. Update values l_1 and l_2.
7: Set $i = i - 1$
8: **if** i = 0 **then**
9: **return** N_1
10: **else**
11: Goto Step (3)
12: **end if**

An heuristic explanation of algorithm 3 is - One starts with a trivial representation of the points, $\mathbf{I}_n$ and at each stage, iteratively refines the representation based on the solutions of the optimization problem at that stage.

4 Implementation

Although algorithm 3 is implementable, note that every stage of the algorithm involves several matrix manipulations and eigenvector calculations. Since eigenvector calculations are not so robust, this algorithm does not work well in practice. Algorithm 4 provides an an efficient alternative of the algorithm 3.

Recall that a connected component of a graph $\mathcal{G}$ is the maximal subgraph of $\mathcal{G}$ which is connected. Note that in the first iteration of algorithm 3 the eigenvectors in step 4 are the indicators of the connected components. This property holds true for next iterations until the number of components are greater than the required number of clusters. This is formalized in proposition 2.

Proposition 2. *Let $\mathcal{G}$ be a similarity graph. At a given level j, let $\{C_1, C_2, \cdots C_l\}$ be the connected components of the graph $\mathcal{G}_{\geq w_j}$. And l is greater than or equal to the required number of connected components. Also let C be the matrix obtained by stacking the vectors $\mathbf{1}_{C_i}/|C_i|$ in columns. Then*

(a) $Tr(C^t L_i C) = 0$ for all $i > j$
(b) Any solution to the optimization problem (22) *at level j is of the form CY where Y is any orthogonal matrix.*

Proposition 2 allows us to optimize the first steps of algorithm 3 by considering the connected components instead of calculating the eigenvectors.

Recall that a discretization scheme (definition 1) was considered in calculating the $\Gamma-$limit. By suitably altering the discretization scheme, one can assume without loss of generality that $\mathcal{G}_{\geq w_2}$ has at least m (required number of clusters) connected components, and that $\mathcal{G}_{\geq w_1}$ has exactly one connected component.

(**Remark** : In practice, this is assured by taking the union of all classes below the threshold. For instance, suppose initial weights considered are $0 \leq w_1 < w_2 < \cdots < w_k \leq 1$. Assume that if $\mathcal{G}_{\geq w_i}$ has at least m components and $\mathcal{G}_{\geq w_{i-1}}$ has less than m components. Then one can reorganize the weights to be $0 \leq w_1 < w_i < \cdots < w_k \leq 1$. This validates the above assumption.)

Algorithm 4 Simplified Efficient algorithm to compute $\Gamma-$limit for ratio-cut.

Input: A weighted graph, $\mathcal{G}$, with bucketed weights $w_1 < w_2 < \cdots < w_j$. Number of clusters required - m.

Output: N - A representation of the subspace spanned by the $\Gamma-$limit of the minimizers.

1: Set $i = k$.
2: **while** Number of connected components of $\mathcal{G}_{\geq w_i}$ is greater than or equal to m **do**
3: Set $i = i - 1$ {We refer to this as an MST-Phase}
4: **end while**
5: Let $\{C_i\}$, $i \in \{1, 2, \cdots, n_c\}$ be the connected components in $\mathcal{G}_{\geq w_i}$.
6: Let I_{C_i} be the vector

$$I_{C_i}(x) = \begin{cases} 1/\sqrt{|C_i|} & \text{if } x \in C_i \\ 0 & \text{othwerwise} \end{cases} \tag{27}$$

7: Construct the matrix N with I_{C_i} as the column vectors.
8: Let $\mathcal{G}_1$ be the graph with the vertex set same as $\mathcal{G}$. Let L_1 be the corresponding Laplacian.
9: Let $\overline{L}_1$ given by $N^t \times L_1 \times N$.
10: Calculate the first m eigenvectors of $\overline{L}_1$ and construct A using these eigenvectors as columns.
11: return $N \times A$.

Thus we have the efficient algorithm 4. In algorithm 4, one simply thresholds the graph to obtain connected components[1]. The matrix N is constructed, whose columns are indicator vectors of the connected components. It is then easy to see that the output of algorithm 4 is a solution to the optimization problem

$$\begin{aligned} \underset{H \in \mathbb{R}^{n \times k}}{\text{minimize}} \quad & Tr(H^t L H) \\ \text{subject to} \quad & H^t H = I \\ & H = NX \text{ for some } X \end{aligned} \tag{28}$$

Thus, another interpretation of the $\Gamma-$limit is - It is a solution to the ratio cut optimization problem with an additional constraint of belonging to the column space of N. (**Remark:** Observe that the matrix NN^t is a projection matrix onto this subspace.) We refer to the output of the algorithm 4 as *Power Ratio cut* or shortly *PRcut*.

Time Complexity: Let n be the number of data points. Assume that $|V| \sim \mathcal{O}(n)$ and $|E| \sim \mathcal{O}(n)$. Observe that steps 1-4 mimics the construction of minimum spanning tree [25], and hence is $\mathcal{O}(n \times log(n))$. In steps 5-7, observe that we are constructing a sparse matrix and hence is $\mathcal{O}(n_c)$. Since N and L_1 are sparse matrices, step 9 is $\mathcal{O}(n)$. Step 10, assuming the worst case scenario of $\overline{L_1}$ being dense matrix, is $\mathcal{O}((n_c)^3)$. Thus the complexity of algorithm 4 is

$$\mathcal{O}(n \times log(n)) + \mathcal{O}((n_c)^3) \tag{29}$$

Under the additional assumptions that $n_c \sim \mathcal{O}(m)$ and $m << n$ we have that the complexity of algorithm 4 is $\mathcal{O}(n \times log(n))$.

Remark: In the trivial case of considering the discretized weights as all equal to 1, we have that the PRcut is identical to Ratio cut.

5 Applications and Analysis

It is clear from above that PRcut is an amalgam of MST based clustering and spectral clustering. It combines 'good' properties of both these methods. In particular it is faster than spectral clustering and more accurate than MST based clustering. All the eigenvector computations are carried out using the SciPy sparse library [26]. The code for generating the results used in this article is available at [27].

Since PRcut is designed to work on weights which take a discrete set of value, one needs a discretization scheme. This discretization scheme is sometimes referred to as *bucketing of weights*. In this article we use *K-means based bucketing*. Observe that the problem of bucketing weights can be rephrased as - combine the weights into buckets so that weights within a bucket are 'alike'. This is once again a clustering problem in itself and hence one can use any clustering method to cluster the weights. K-means is a simple choice known for its efficiency and simplicity. In this case each weight is represented by the mean of the cluster to which the weight belongs to.

5.1 Comparison with MST based clustering

Compared to MST based clustering PRcut does not break ties arbitrarily. Instead PRcut takes into consideration the sizes of the clusters. For example consider a simple graph as in figure 1(a). The solid black lines indicates a choice of MST constructed. Figure 1(b) indicates a solution obtained by MST based clustering. Given the MST from figure 1(a), since MST based clustering picks the edge randomly, there is $1/2$ chance that figure 1(b) is obtained as a solution. On the other hand, since PRcut takes into consideration the sizes of the clusters, only the solution in figure 1(c) is obtained. Thus PRcut performs better in such cases.

5.2 Comparison with spectral clustering

Notably, PRcut preserves several good properties of spectral clustering as well. In particular, it preserves the property of being able to discover non globular clusters. For instance consider the data as in figure 2(a). Figure 2(b) indicates the results obtained using the spectral clustering method. Same

1. Thresholding a graph to obtain connected components is equivalent to constructing an MST and removing the lower weight edges. Hence we refer to this as MST phase.

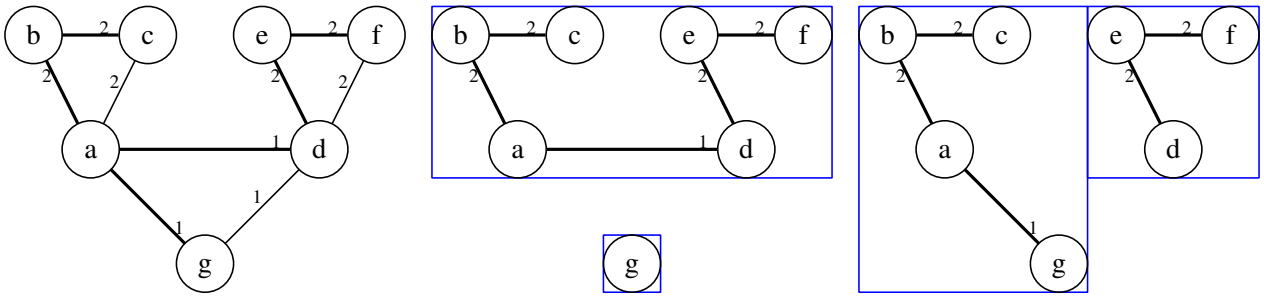


Fig. 1. Simple example illustrating the difference between PRcut and MST based clustering. (a) An example graph. Solid lines indicates a maximum spanning tree. (b) A result obtained using MST based clustering. (c) Result obtained using PRcut. Observe that PRcut provides better results compared to simple MST based clustering.

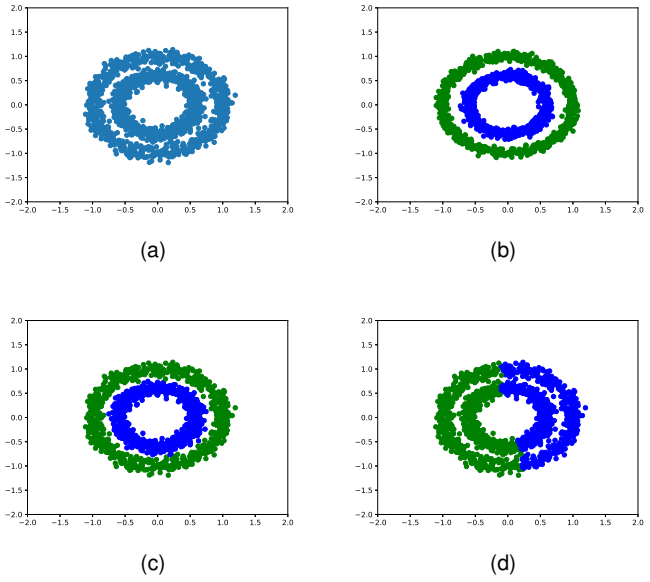


Fig. 2. Toy example illustrating the robustness of PRcut when compared to Ratio cut (Rcut). (a) Two circles dataset [28] with noise=0.07. (b) Result obtained using Ratio cut on the two circles dataset with noise=0.05 .Same result is obtained with PRcut as well. (c) Result obtained using PRcut on two circles dataset with noise=0.07. (d) Result obtained using Ratio cut on two circles dataset with noise=0.07.

result is obtained using Power Ratio cut as well. However, as the noise increases, spectral clustering breaks down (figure 2(d)) while PRcut is slightly more robust (figure 2(c)). This is also illustrated in figure 3.

Another important property of spectral clustering methods is that they penalize dissimilar sized clusters. This effect is preserved for PRcut as well. To verify this we conduct the following experiment - Consider the image in figure 4(a). The image is divided into 3 components - black component on the left, white component on the right and the boundary component in between. This is a synthetic example of the constant gradient at boundaries between classes, referred to as *flatzones*. Clustering the above image we expect the boundary component to be split into two parts - one corresponds to the black component and another corresponds to the white component. Figure 4(b) and figure 4(c) illustrate the result obtained with PRcut and Ratio cut.

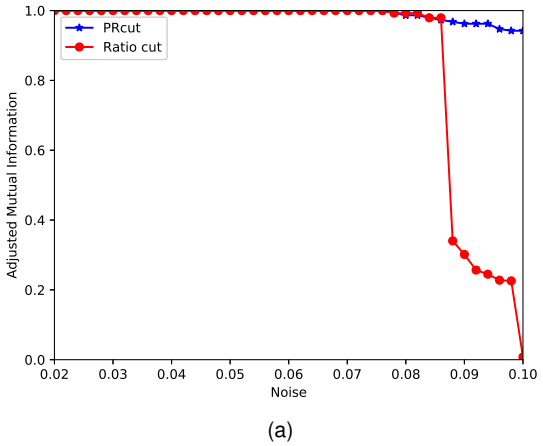


Fig. 3. Plot of the adjusted mutual information plotted vs the noise in the two circles dataset [28]. The values are averaged over 10 iterations. Note that at higher noise levels, PRcut performs better than Ratio cut.

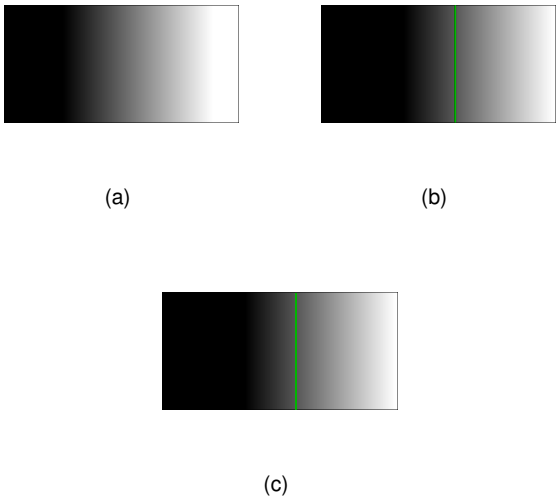
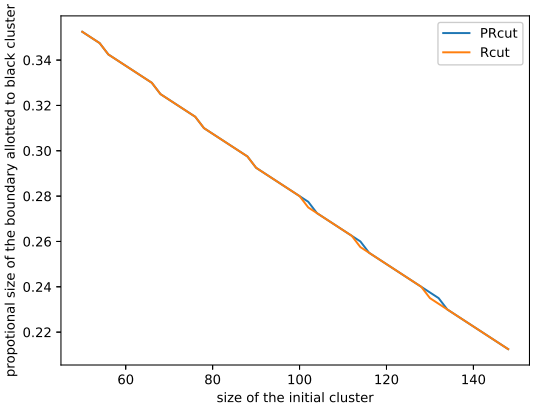
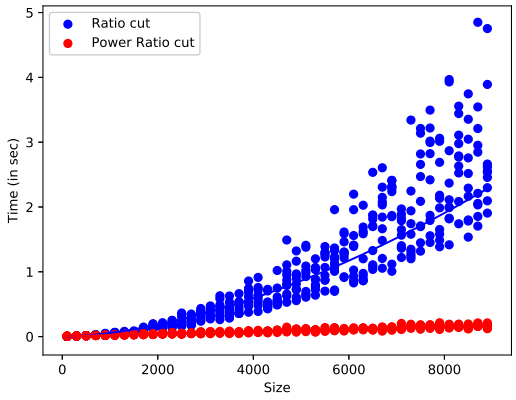


Fig. 4. (a) A synthetic example of the of a ramp image (constant horizontal gradient). (b) Result obtained using PRcut (c) Result obtained using Rcut. Observe that PRcut and Rcut gives similar partitions.



(a)

Fig. 5. Plot to illustrate that both PRcut and Rcut penalize differently sized clusters in the same way. We change the size of black component in figure 4(a) initially before clustering and plot the size of the boundary allotted to the black component after clustering. X-axis indicates the size of the black component initially before clustering. Y-axis indicates the size of the boundary allotted to the black component. Note that, for both PRcut and Rcut, the size of the boundary component allotted to the black component reduces as the initial size of the black component increases. This allows us to conclude that PRcut penalizes dissimilar clusters as well.



(a)

Fig. 6. Time complexity of PRcut vs Rcut as a function of data size on blobs dataset [29] with parameters - $n\_features = 2$, $centers = 2$. Time is measured in seconds. Observe that for small data sizes the difference between PRcut and Rcut is not significant, while for large data sizes it varies considerably. The variance in the timing of the ratio cuts is due to the optimizations of the SciPy sparse library [26].

As the initial size of the black component varies, due to the above property of spectral clustering, we expect the amount of "boundary component" allotted to the black component to reduce. This is verified in figure 5 for Rcut. This is also the case for PRcut as well as shown in figure 5. Hence, one can conclude that PRcut preserves the property of penalizing dissimilar sized clusters.

The main difference between spectral clustering methods and PRcut is in the case of run time. Due to preprocessing the data using MST, the size of the data is reduced drastically and hence computing time is saved. As discussed earlier, under some conditions the running time of PRcut increases as $\mathcal{O}(n \times log(n))$ while spectral clustering methods take $\mathcal{O}(n^{3/2})$ [5]. This is illustrated in figure 6. Note that for small problems the difference performance of PRcut and Rcut is negligible. However, as the size of the problem increases, PRcut performs better than Rcut.

5.3 Multiscale Combinatorial Grouping (Segmentation)

One of the main applications of spectral clustering is in the domain of image segmentation. Normalized cuts have been widely used to segment an image into meaningful regions [5], [14], [31], [32]. Normalized cuts is used to globalize the local information. In [14], the authors propose a sequence of steps to obtain segmentation of the images. One of the most important steps in the pipeline is normalized cut. The following lines are taken from [14] -

> The normalized cuts criterion is a key globalization mechanism of recent high-performance contour detectors $\cdots$. Although powerful, such spectral graph partitioning has a significant computational cost and memory footprint that limit its scalability.

In this section, we use exactly the same pipeline as that in [14], but instead of using spectral clustering, we use PRcut.

TABLE 1
Table indicating p-values on BSDS dataset under the hypothesis that the average F_{op} and F_b scores of Ncut and PRcut are equal.

(p - values)	F_{op}	F_b
PRcut vs Ncut	0.2902	0.2819
dPRcut vs dNcut	0.3499	0.6884

The authors then proposed a heuristic method to calculate the eigenvectors faster, termed as *dNcut*. This method reduces the eigenvalue problem and hence is faster and has less memory requirements, a technique similar to Nystrom's method [12], [33].

Since, heuristically PRcut works by reducing unnecessary computation, PRcut is compatible with other reduction techniques. Hence using the same reduction technique as in [14], we have another technique called *dPRcut.* Also, all these methods have a multiscale version as well.

These methods were experimentally compared using the BSDS500 dataset [32]. Few selected contour saliency maps [34] (also referred to as UCM, ultrametric contour maps in [35]) are shown in figure 7. The precision recall curves were plotted using the techniques described in [30], shown in figure 8. From this it can be seen that the results do not differ by much. Scatterplots in figures 8 (c), (d), (e), (f) corroborate this observation. Under the null hypothesis that the methods are equivalent, the p-values calculated using the t-test are given in table 1. It is clear that there is *no* sufficient evidence to claim that one method is superior to the other in terms of results. Note that dPRcut does not vary much compared to dNcut, thus establishing that PRcut is compatible with other reduction techniques as well.

However, PRcut is much faster than the Ncut as shown in figure 9. In fact, the time of PRcut is comparable to $dNcut$ as well. Using the extra layer of approximation in $dPRcut$ and $dNcut$ did not show much of a difference since the size of the images is limited.

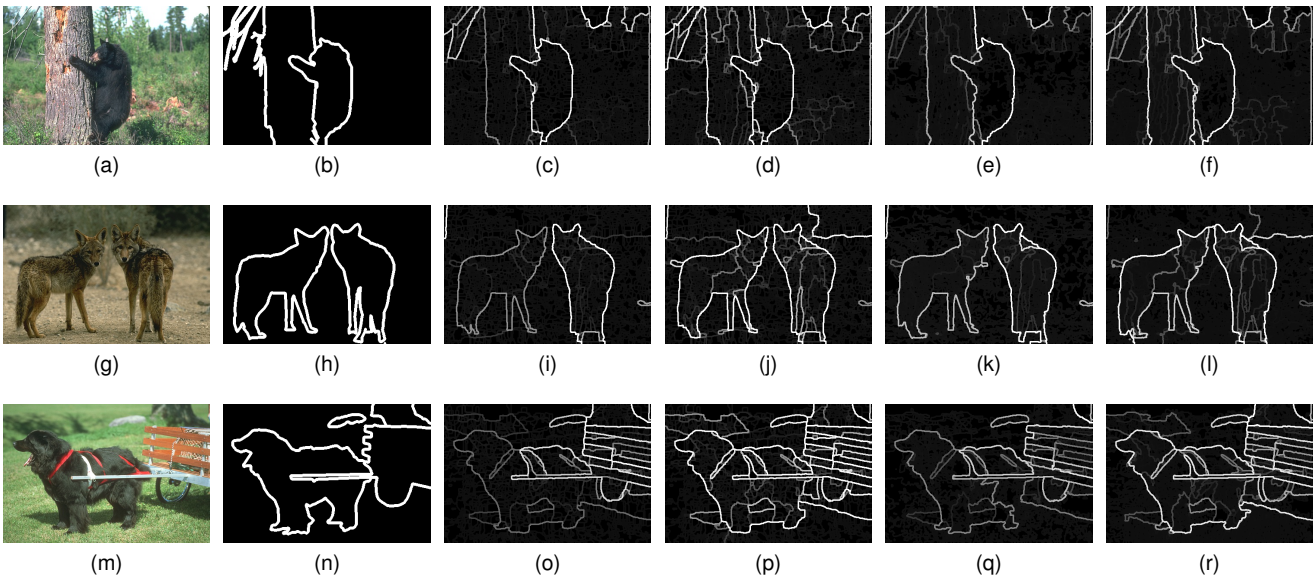


Fig. 7. Selected Results from BSDS500. First-column: Original Image, Second-Column: Groundtruth, Third-Column: PRcut, Fourth- Column: Normalized cut, Fifth-Column : Multiscale PRcut, Sixth-Column : Multiscale Ncut

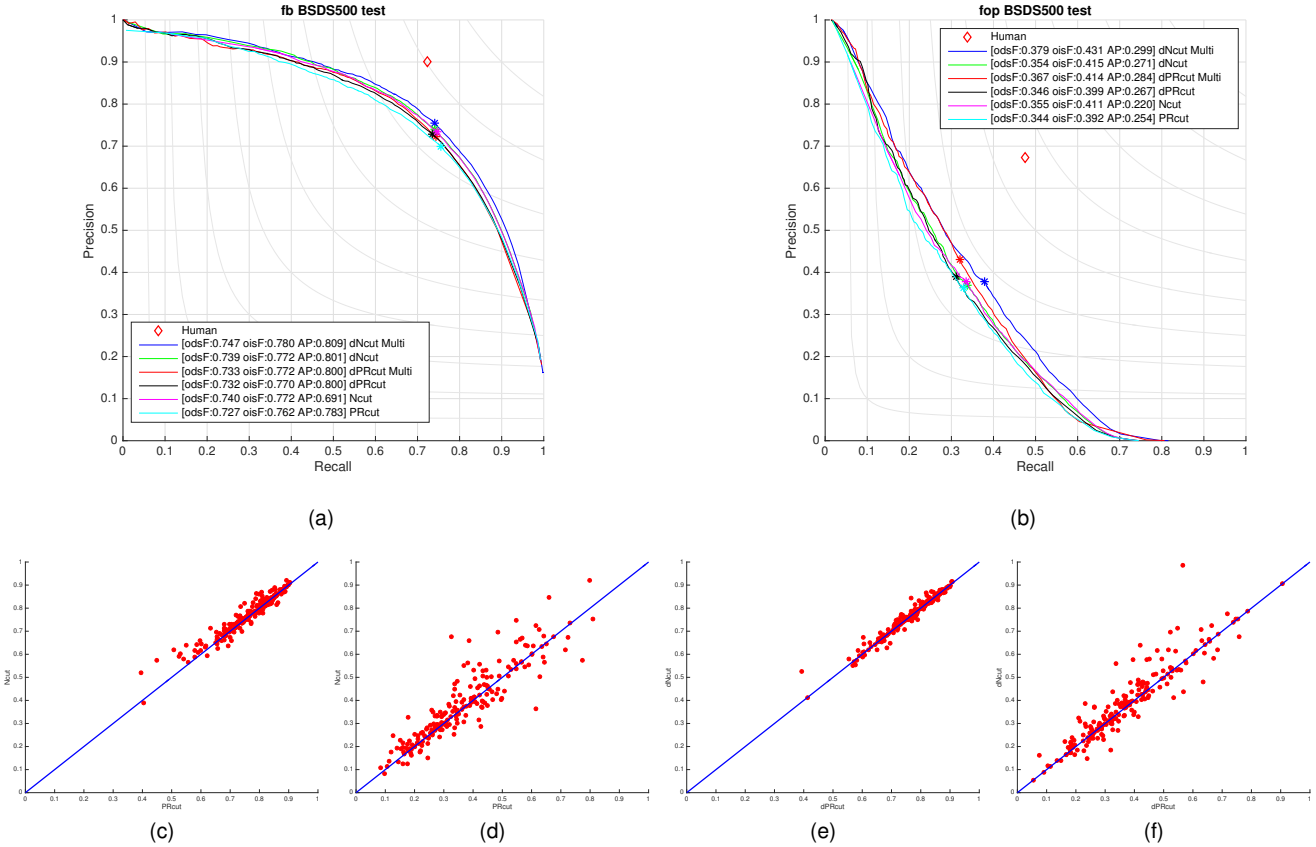
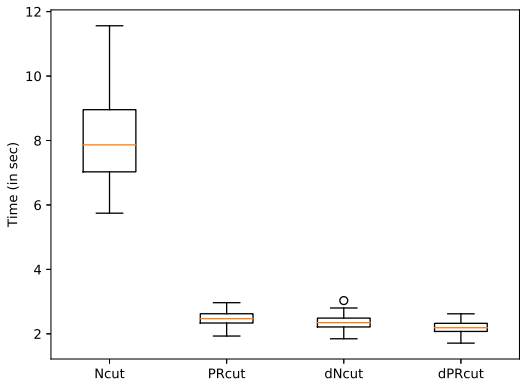
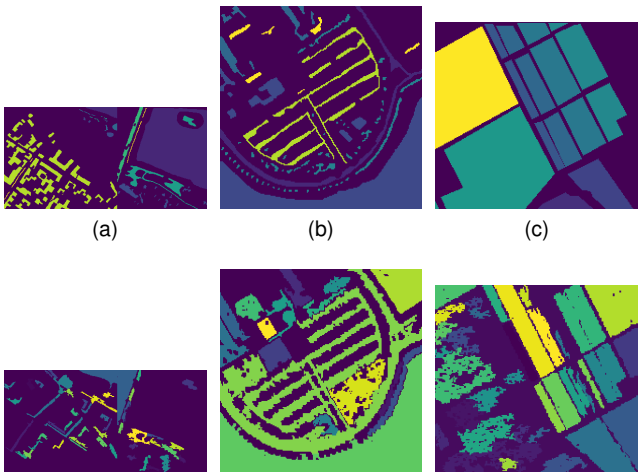


Fig. 8. (a) Shows the precision-recall curves of the measure F_b as defined in [30]. (b) Shows the precision-recall curves of the measure F_{op} as defined in [30]. 'dPRcut Multi' and 'dNcut Multi' indicate the multiscale versions of PRcut and Ncut respectively. (c) Scatter plot of optimal F_b measure on individual images between PRcut and Ncut. (c) Scatter plot of optimal F_{op} measure on individual images between PRcut and Ncut. (e) Scatter plot of optimal F_b measure on individual images between dPRcut and dNcut. (f) Scatter plot of optimal F_{op} measure on individual images between dPRcut and dNcut. Observe from (c)-(f) that there is no major difference in accuracy between PRcut and Ncut measures. All results are calculated on BSDS500 dataset with the MCG algorithm, using either a NCut or a PRCut step.



(a)

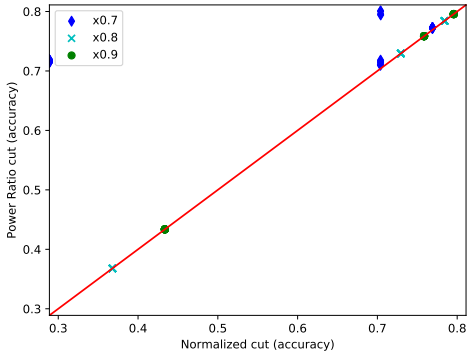
Fig. 9. Box plot of the time taken by various methods. The time is on Y-axis and is measured in seconds. The time taken is calculated from the input to the final segmented output obtained by MCG on the BSDS500 dataset.



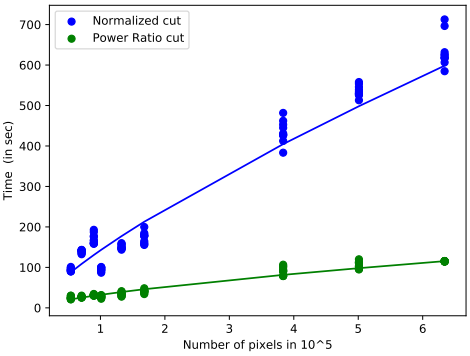
(a) (b) (c)

(d) (e) (f)

Fig. 10. Top row: Groundtruth images of Pavia Center, Pavia University and Salinas hyperspectral images. Bottom row: Results using PRcut of Pavia Center, Pavia University and Salinas hyperspectral images



(a)



(b)

Fig. 11. (a) Scatter plot of the accuracy of normalized cuts vs power ratio cuts on hyperspectral data. The red line indicates the line $x = y$. As one can see, all the points lie above or on the line $x == y$. Hence power ratio cut results are better than normalized cuts (or at least equal). (b) Plots of executing times vs the number of pixels in the image. It is easy to observe that PRcut scales much better compared to normalized cuts.

5.4 Hyperspectral Data

Clustering is also referred to as *unsupervised learning*. Solutions to clustering problem, unlike supervised learning, cannot offer black box solutions [7], [36]. It is usually the case that several aspects of clustering are dictated by the domain knowledge. One of the main advantages of PRcut is that it offers better control over the clustering procedure than spectral clustering. We now describe the application of spectral clustering to *hyperspectral data*. These results are first reported in [37].

A feature of the hyperspectral data is that most of the data remains unclassified (class 0). Thus the classes are not balanced and hence spectral clustering methods cannot be used directly. Moreover, spectral clustering methods cannot be easily adapted to such cases as well. However, since PRcut has two phases - MST phase and spectral clustering, one can suitably modify the algorithm to requirements. In the case of hyperspectral data, this is obtained by ignoring the small clusters after the MST phase in PRcut (explained in detail in [38]). This allows to obtain higher accuracy with PRcut compared to Rcut, as observed in case of Salinas dataset in table 2.

We have considered three hyperspectral datasets - (a) Salinas (512×217) (b) Pavia University (610×340) (c) Pavia Center (1096×715). For each of the datasets, we have considered three ratios - $0.7, 0.8, 0.9$, of their sizes. Each experiment was repeated 10 times and have taken the average. Also, the datasets were over segmented and the accuracy is calculated by assuming that each cluster is labelled with the largest groundtruth label. That is, let $C = \{C_1, C_2, \cdots, C_n\}$ be classes obtained by clustering and let $C^* = \{C_1^*, C_2^*, \cdots, C_m^*\}$ indicate the groundtruth classes. Let $N(C_i, C_j^*)$ be the number of pixels labelled C_i and C_j^*. The accuracy is then given by

$$accuracy(C, C^*) = \frac{\sum_i (\max_j N(C_i, C_j^*))}{\text{Total number of pixels}} \qquad (30)$$

The timing is measured in seconds. All experiments are done on *Intel(R) Xeon(R) CPU E5620 at 2.40GHz* with RAM size of 16 GigaBytes.

An example of the results obtained of the PRcut on subsets of these images and their groundtruth is shown in

TABLE 2
Table indicating the average times and accuracy of Ratio cut and Power Ratio cut on hyperspectral data. The timing is reported in seconds. The accuracy is measured by (30).

	Data	Process Time	Rcut		PRcut	
			Time	Accuracy	Time	Accuracy
Ratio = 0.7	Pavia University	129.33	95.15	0.70	27.38	0.74
	Pavia Center	530.95	435.01	0.76	91.38	0.77
	Salinas	142.59	94.27	0.28	24.92	0.71
Ratio = 0.8	Pavia University	194.14	151.00	0.72	34.06	0.78
	Pavia Center	623.74	535.79	0.78	104.46	0.78
	Salinas	202.98	139.31	0.36	26.88	0.74
Ratio = 0.9	Pavia University	279.42	163.66	0.75	40.71	0.76
	Pavia Center	830.18	633.638	0.79	115.09	0.79
	Salinas	271.56	170.28	0.43	31.63	0.78

figure 10. Average of accuracy and times for each of the methods is shown in table 2. Scatter plots of accuracy and times are also plotted in figure 11. It is easy to see from the table that PRcut is faster and gives better accuracy than ratio cuts (or at least equal).

5.5 High dimensional data

Another characteristic of the data is its dimensionality. In high dimensions, due to the curse of dimensionality [41], [42], it is difficult to measure the distances accurately and hence difficult to construct an edge weighted graph. The distance between points become unreliable in higher dimensions. Thus spectral clustering cannot be directly applied for clustering high dimensional data. Usually a preprocessing step is applied to the data to reduce the dimension, such as principal component analysis (PCA). The reduced data is then used for further processing.

Here the MNIST dataset [43] is used to experiment with high dimensions. The number of data points are 42000 and each data point has a dimension of 784. The dataset if first processed with PCA to reduce the dimensionality. Figure 12 shows the results on the selected classes. Table 3 shows the results obtained with PRcut and spectral clustering on randomized samples of the datasets, with varying dimension. It is clear from these results PRcut is faster than Rcut while preserving the precision of the results.

6 Conclusion

To summarize, we have proposed a faster alternative to the spectral clustering - Power Ratio cut or PRcut. This was obtained by considering the $\Gamma-$limit of the spectral clustering. Intuitively PRcut uses maximum spanning tree to reduce the size of the dataset and then solves the appropriate eigenvalue problem. This results in a much faster algorithm than spectral clustering. Also, by considering the $\Gamma-$limit several good properties of spectral clustering such as - penalizing unequal cluster distribution, ability to detect non-convex clusters is preserved. PRcut is compared with both MST based clustering and spectral clustering considering several toy examples. Its efficiency as an alternate to Ncut is exhibited by suitably replacing Ncut in the MCG pipeline with PRcut and comparing the results on BSDS. Results of comparison with Ncut on hyperspectral datasets and MNIST dataset are also provided. The code to generate the results in this paper is available at [27].

Appendix

Proof of Theorem 2

Proof. We first show that any matrix H of the form AKY is a solution to (7). We have

$$\begin{aligned} Tr((AKY)^t L(AKY)) &= Tr(Y^t(AK)^t L(AK)Y) \\ &= Tr((AK)^t L(AK)) \\ &= Tr(K^t(A^t LA)K) \end{aligned}$$

since Y is an orthogonal matrix. Now, $A^t LA$ is of the form

$$\begin{bmatrix} \lambda_1 & 0 & \cdots & 0 \\ 0 & \lambda_2 & \cdots & 0 \\ 0 & \cdots & \cdots & 0 \\ 0 & 0 & \cdots & \lambda_m \end{bmatrix}$$

So, we have

$$\begin{aligned} Tr(K^t(A^t LA)K) &= \lambda_1 + \lambda_2 + \cdots + \lambda_{l_1} + \lambda_{(m)} Tr(X^t IX) \\ &= \lambda_1 + \lambda_2 + \cdots + \lambda_m \\ &= \min Tr(H^t LH) \end{aligned}$$

Since, the minimum value of $Tr(H^t LH)$ is equal to $\lambda_1 + \lambda_2 + \cdots + \lambda_m$ [24]. To show the other side, note that the set of all the solutions of (7) is

$$\{H \in \mathbb{R}^{n\times m} \mid Tr(H^t LH) = \lambda_1 + \lambda_2 + \cdots + \lambda_m\}$$

Let $\overline{A} = [e_1, e_2, \cdots e_n]$, i.e the matrix obtained by stacking all the eigenvectors in columns. Then any H can be written as $\overline{A}Z$ where $Z^t Z = \mathbf{I}$. Now, note that since $\{\lambda_1, \lambda_2, \cdots, \lambda_n\}$ can be arbitrary, we need to have that

$$Z_{ij} = 0 \text{ if } \lambda_i > \lambda_{(m)}.$$

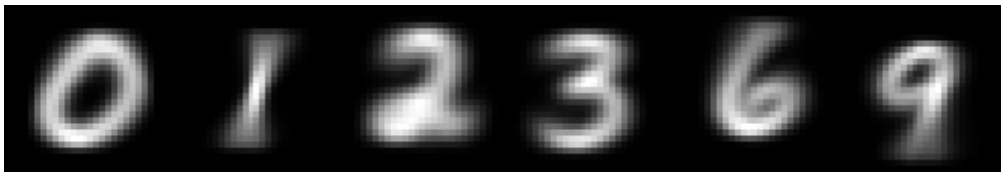
Thus we can ignore the lower part of matrix Z. If $A = [[e_1, e_2, \cdots e_l]]$, then we have that H is of the form AZ where Z is a $l \times m$ matrix such that $Z^t Z = \mathbf{I}$. Now let,

$$Z = \begin{bmatrix} Z_{l_1,l_1} & Z_{l_1,m-l_1} \\ Z_{l_2,l_1} & Z_{l_2,m-l_1} \end{bmatrix}$$

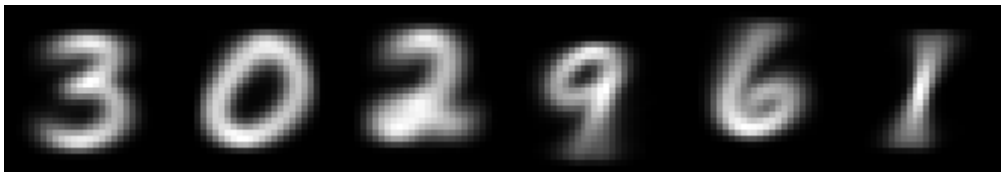
Also, note that $A^t LA$ is of the form

$$\begin{bmatrix} \lambda_1 & 0 & \cdots & 0 \\ 0 & \lambda_2 & \cdots & 0 \\ 0 & \cdots & \cdots & 0 \\ 0 & 0 & \cdots & \lambda_l \end{bmatrix}$$

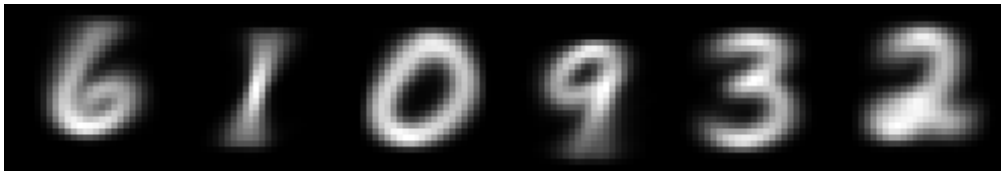
Since Z should satisfy $Tr(Z^t A^t LAZ) = Tr(H^t LH) = \lambda_1 + \lambda_2 + \cdots + \lambda_m$, we need to have that Z_{l_1,l_1} is full rank, and $Z^t_{l_1,l_1} Z_{l_1,l_1} = \mathbf{I}$. Hence Z must be of the form KY where K is a matrix of the form (23) and Y is any orthogonal matrix. Hence all the solutions to (7) must be of the form AKY. □



(a)



(b)



(c)

Fig. 12. Results obtained on selected classes of the MNIST dataset. (a) Average of the groundtruth classes. (b) Average of clusters obtained by Rcut (c) Average of clusters obtained by PRcut.

TABLE 3
Table indicating the average times and accuracy of Ratio cut and Power Ratio cut on MNIST dataset. The timing is reported in seconds. The accuracy is measured by (30). ARI indicates Adjusted Rand Index [39], and AMI indicates Adjusted Mutual Information [40].

	Data Size	Process Time	Rcut				PRcut			
			Time	Accuracy	ARI	AMI	Time	Accuracy	ARI	AMI
Dim = 10	4318	0.56	2.03	0.54	0.40	0.52	1.33	0.54	0.40	0.51
	8489	1.29	15.36	0.56	0.42	0.54	7.48	0.56	0.42	0.54
	12634	2.17	33.61	0.56	0.43	0.55	26.11	0.56	0.43	0.55
Dim = 15	4318	0.73	2.03	0.56	0.45	0.57	1.38	0.56	0.45	0.57
	8489	1.97	12.52	0.57	0.46	0.59	7.90	0.57	0.47	0.59
	12634	3.18	36.13	0.58	0.49	0.61	25.49	0.58	0.49	0.61
Dim = 20	4318	0.94	2.86	0.57	0.46	0.58	1.87	0.57	0.46	0.58
	8489	2.96	11.96	0.58	0.49	0.61	9.84	0.58	0.49	0.61
	12634	4.34	32.66	0.59	0.51	0.63	24.23	0.59	0.51	0.63

PROOF OF LEMMA 1

Proof. One side follows from the relation

$$\mathcal{H}(\mathcal{M}(H')) \supseteq H'$$

For the other side, let H be some matrix which spans the same subspace as $\hat{H} = A_k KY$. Then there exists an orthogonal matrix Q such that $HQ = \hat{H}$. Hence $H = \hat{H}Q^t = A_k KYQ^t$. This is still in the form $A_k KY$ where Y is some orhogonal matrix. □

PROOF OF PROPOSITION 2

Proof. To prove (a), note that since C_i is a connected component of $\mathcal{G}_{\geq w_j}$, it is also a union of connected components of $\mathcal{G}_{\geq w_i}$ for all $i >= j$. Also, we know that if L denotes a Laplacian of the graph, then $L\mathbf{1}_{C_i} = 0$ if C_i is a connected component in the graph. Hence proved.

Since (a) is true, we know that C is a solution to (22) at level j. Thus all matrices of the form CY are solutions. Now, since $Tr(C^t L_i C) = 0$ for all $i >= j$, any vector c belonging to the column space of the solution must satisfy $c^t L_i c = 0$ for all $i >= j$. This implies that c belongs to the column space of the 0 eigenvectors, which are indicators of

connected components. Hence c must belong to the column space of the indicators of connected components for each $\mathcal{G}_i$, $i >= j$. This implies that c belongs to the column space of indicators of connected components of $\mathcal{G}_{\geq w_j}$, and hence belongs to the column space of C. Hence proved. □

ACKNOWLEDGMENTS

AC and SD would like to thank Indian Statistical Institute. LN would like to acknowledge the funding received from - ANR-15-CE40-0006 CoMeDiC, ANR-14-CE27-0001 GRAPH-SIP research grants and Programme d'Investissements d'Avenir (LabEx BEZOUT ANR-10-LABX-58). BSDS would like to acknowledge the support received from the Science and Engineering Research Board (SERB) of the Department of Science and Technology (DST) with the grant number EMR/2015/000853, and the Indian Space Research Organization (ISRO) with the grant number ISRO/SSPO/Ch-1/2016-17.

REFERENCES

[1] M. Iwayama and T. Tokunaga, "Cluster-based text categorization: a comparison of category search strategies," in *Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 1995, pp. 273–280.

[2] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM computing surveys (CSUR)*, vol. 41, no. 3, p. 15, 2009.

[3] G. Punj and D. W. Stewart, "Cluster analysis in marketing research: Review and suggestions for application," *Journal of marketing research*, pp. 134–148, 1983.

[4] A. K. Jain, "Data clustering: 50 years beyond k-means," *Pattern recognition letters*, vol. 31, no. 8, pp. 651–666, 2010.

[5] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 22, no. 8, pp. 888–905, 2000.

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

[7] A. K. Jain, M. N. Murty, and P. J. Flynn, "Data clustering: a review," *ACM computing surveys (CSUR)*, vol. 31, no. 3, pp. 264–323, 1999.

[8] U. Von Luxburg, "A tutorial on spectral clustering," *Statistics and computing*, vol. 17, no. 4, pp. 395–416, 2007.

[9] D. Yan, L. Huang, and M. I. Jordan, "Fast approximate spectral clustering," in *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2009, pp. 907–916.

[10] J. Chen, H.-r. Fang, and Y. Saad, "Fast approximate knn graph construction for high dimensional data via recursive lanczos bisection," *Journal of Machine Learning Research*, vol. 10, no. Sep, pp. 1989–2012, 2009.

[11] 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*. ACM, 2004, pp. 551–556.

[12] C. Fowlkes, S. Belongie, F. Chung, and J. Malik, "Spectral grouping using the nystrom method," *IEEE transactions on pattern analysis and machine intelligence*, vol. 26, no. 2, pp. 214–225, 2004.

[13] Y. Song, W.-Y. Chen, H. Bai, C.-J. Lin, and E. Chang, "Parallel spectral clustering," *Machine Learning and Knowledge Discovery in Databases*, pp. 374–389, 2008.

[14] J. Pont-Tuset, P. Arbelaez, J. T. Barron, F. Marques, and J. Malik, "Multiscale combinatorial grouping for image segmentation and object proposal generation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 1, pp. 128–140, 2017.

[15] C. T. Zahn, "Graph-theoretical methods for detecting and describing gestalt clusters," *IEEE Transactions on computers*, vol. 100, no. 1, pp. 68–86, 1971.

[16] A. Braides, *Gamma-convergence for Beginners*. Clarendon Press, 2002, vol. 22.

[17] C. Couprie, L. Grady, L. Najman, and H. Talbot, "Power watershed: A unifying graph-based optimization framework," *IEEE transactions on pattern analysis and machine intelligence*, vol. 33, no. 7, pp. 1384–1399, 2011.

[18] A. K. Sinop and L. Grady, "A seeded image segmentation framework unifying graph cuts and random walker which yields a new algorithm," in *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*. IEEE, 2007, pp. 1–8.

[19] J. Cousty, G. Bertrand, L. Najman, and M. Couprie, "Watershed cuts: Minimum spanning forests and the drop of water principle," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 8, pp. 1362–1374, 2009.

[20] ——, "Watershed cuts: Thinnings, shortest path forests, and topological watersheds," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 5, pp. 925–939, 2010.

[21] L. Najman, "Extending the PowerWatershed framework thanks to Γ-convergence," *SIAM Journal on Imagine Sciences*, vol. 10, no. 4, pp. 2275–2292, Nov. 2017. [Online]. Available: https://hal.archives-ouvertes.fr/hal-01428875

[22] A. Y. Ng, M. I. Jordan, Y. Weiss *et al.*, "On spectral clustering: Analysis and an algorithm," *Advances in neural information processing systems*, vol. 2, pp. 849–856, 2002.

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

[24] H. Lütkepohl, *Handbook of Matrices*, 1st ed. Wiley, 1997.

[25] J. Cousty, L. Najman, Y. Kenmochi, and S. Guimarães, "Hierarchical segmentations with graphs: quasi-flat zones, minimum spanning trees, and saliency maps," *Journal of Mathematical Imaging and Vision*, 2017. [Online]. Available: https://hal.archives-ouvertes.fr/hal-01344727

[26] E. Jones, T. Oliphant, P. Peterson *et al.*, "SciPy: Open source scientific tools for Python," 2001–. [Online]. Available: https://docs.scipy.org/doc/scipy/reference/sparse.html

[27] A. Challa, "Power spectral clustering," https://github.com/ac20/Power-Spectral-Clustering, 2018.

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

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

[30] J. Pont-Tuset and F. Marques, "Supervised evaluation of image segmentation and object proposal techniques," *IEEE transactions on pattern analysis and machine intelligence*, vol. 38, no. 7, pp. 1465–1478, 2016.

[31] R. Xiaofeng and L. Bo, "Discriminatively trained sparse code gradients for contour detection," in *Advances in neural information processing systems*, 2012, pp. 584–592.

[32] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik, "Contour detection and hierarchical image segmentation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 33, no. 5, pp. 898–916, 2011.

[33] C. K. Williams and M. Seeger, "Using the nyström method to speed up kernel machines," in *Advances in neural information processing systems*, 2001, pp. 682–688.

[34] L. Najman and M. Schmitt, "Geodesic saliency of watershed contours and hierarchical segmentation," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 18, no. 12, pp. 1163–1173, 1996.

[35] P. Arbelaez, "Boundary extraction in natural images using ultrametric contour maps," in *Computer Vision and Pattern Recognition Workshop, 2006. CVPRW'06. Conference on*. IEEE, 2006, pp. 182–182.

[36] J. M. Kleinberg, "An impossibility theorem for clustering," in *Advances in neural information processing systems*, 2003, pp. 463–470.

[37] A. Challa, S. Danda, B. S. Daya Sagar, and L. Najman, "Power Spectral Clustering on Hyperspectral Data," in *International Geoscience and Remote Sensing Symposium*. Forth Worth, United States: IEEE, Jul. 2017. [Online]. Available: https://hal.archives-ouvertes.fr/hal-01484896

[38] ——, "An Introduction to Gamma-Convergence for Spectral Clustering," in *Discrete Geometry for Computer Imagery*, ser. Lecture Note In Computer Sciences, vol. 10502, Kropatsch, Walter G. and Artner, Nicole M. and Janusch, Ines. Vienna, Austria: Springer, Sep. 2017, pp. 185–196. [Online]. Available: https://hal.archives-ouvertes.fr/hal-01427957

[39] L. Hubert and P. Arabie, "Comparing partitions," *Journal of classification*, vol. 2, no. 1, pp. 193–218, 1985.

[40] N. X. Vinh, J. Epps, and J. Bailey, "Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance," *Journal of Machine Learning Research*, vol. 11, no. Oct, pp. 2837–2854, 2010.

[41] R. E. Bellman, *Adaptive control processes: a guided tour*. Princeton university press, 2015.
[42] J. Friedman, T. Hastie, and R. Tibshirani, *The elements of statistical learning*. Springer series in statistics New York, 2001, vol. 1.
[43] Y. Lecun and C. Cortes, "The MNIST database of handwritten digits." [Online]. Available: http://yann.lecun.com/exdb/mnist/

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 his MSc and PhD degrees in geoengineering and remote sensing from the Faculty of Engineering, Andhra University, Visakhapatnam, India, in 1991 and 1994 respectively. He is also first Head of the SSIU. Earlier, he worked in the 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 2001-2007. Since 2017, he has been a Visiting Professor at the University of Trento, Trento, Italy. His research interests include mathematical morphology, GISci, digital image processing, fractals and multifractals, their applications in extraction, analyses, and modelling 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 Mor-phology in Geomorphology and GISci," CRC Press: Boca Raton, 2013, p. 546. He recently co-edited two special issues on "Filtering and Segmentation with Mathematical Morphology" for IEEE Journal of Selected Topics in Signal Processing (v. 6, no. 7, p. 737-886, 2012), and "Applied Earth Observation and Remote Sensing in India" for IEEE Journal of Selected Topics in Applied Earth Observation and Remote Sensing (v. 10, no. 12, p. 5149-5328, 2017). 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 1995-1996. He received the Dr. Balakrishna Memorial Award from Andhra Pradesh Academy of Sciences in 1995, the Krishnan Gold Medal from Indian Geophysical Union in 2002, and the "Georges Matheron Award-2011 (with Lecturership)" of the International Association for Mathematical Geosciences. 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.

Laurent Najman (SM'17) received the Habilitation à Diriger les Recherches in 2006 from University the 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 Institut Gaspard Monge, Université Paris-Est Marne-la-Vallée. His current research interest is discrete mathematical morphology and discrete optimization.