Power Spectral Clustering

Aditya Challa, Sravan Danda, B 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

***Abstract*—**
The problem of clustering has been an important problem since the early 20th century and several possible solutions were proposed. With the rise of computing machines clustering has become an important part of many data mining tasks, focussed on fast implementations. An important task related to clustering is image segmentation. In the set of solutions to the clustering problem, the method of spectral clustering has obtained wide interest due to its ability to detect non-convex clusters in the data. In this article, we propose a fast alternative to the spectral clustering, obtained by taking the $\Gamma-$limit. We explore the links between the new method and MST based clustering. We then show that the proposed method is as good as the spectral clustering with the help of experiments on several datasets. We also show that the new method is scalable to large data unlike the classical spectral clustering methods.

***Index Terms*—Image Interpolation, Mathematical Morphology, Morphological Interpolation.**

I. Introduction

The problem of clustering can be stated as - given a set of data $\{x_i\}$, partition the set into groups such that elements in the same set are 'similar' and elements belonging to the different sets are 'dissimilar'. This has been a problem of interest since the early 20th century and is widely used in several areas of application. The vast literature available on the topic of clustering shows the extent of applications for a solution to the clustering problem. An important problem in computer vision - Image segmentation can be formulated as a clustering problem [1], [2]. Clustering documents is also used in searching for similar documents of interest [3], anomaly detection [4], and market research [5]. A comprehensive monograph of various aspects of clustering can be found in [6], [7], [1].

A specific 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, spectral clustering methods are able to detect the non-convex clusters and work well in high dimensions. These methods construct a graph from the data and use the eigenvectors of the laplacian of the graph constructed from the data. However the calculation of the eigenvectors, which is computationally hard and prone to errors. Several efforts were made to increase the speed and accuracy of the spectral clustering methods, including parallelizing [9]. In [10] the authors propose to reduce the size of the similarity matrix to speed up the normalized cuts procedure for image segmentation [2].

Aditya Challa, Sravan Danda and B.S.Daya Sagar are with the Systems Science and Informatics Unit, Indian Statistical Institute, Bangalore.

Laurent Najman is with Université Paris-Est, LIGM, Equipe A3SI, ESIEE, France.

In [11] the authors proposed to take the limit of the minimizers of the energy function in [12] called *power watershed*. This is referred to as the $\Gamma-$limit. 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 [13]). In [14] 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 calculate the gamma limit of the spectral clustering starting from the generic algorithm given in [14]. We refer to this $\Gamma-$limit as the *power spectral clustering* or *Power Ratio cut* or more simply *PRcut*.

In section II we introduce the background of various concepts used in the rest of the article - spectral clustering, MST clustering and $\Gamma-$convergence. In section III we start with an generic algorithm 1 to calculate the gamma limit, and progressively simplify and characterize different parts of the algorithm to calculate the gamma limit, ending with an implementable version in algorithm 4. In section IV we further increase the efficiency by identifying the commonality between eigenvectors and connected components of the laplacian. In section V we cover various issues which arise in practice due to finite precision and numerical errors of calculating the eigenvalues. In particular, we make few simple and reasonable assumptions so that the algorithm can be implemented robustly in practice.

The main contributions of the article are:

(1) An algorithm to calculate the 'equivalent' $\Gamma-$limit of spectral clustering is proposed.
(2) The algorithm is analyzed and tested on various datasets to show empirically that it gives competing results to other comparable algorithms.
(3) The main problem with spectral clustering is that, specral clustering methods lacks speed (base algorithm being $\mathcal{O}(n^3)$) and scalability. Several attempts were made in improving the speed and scalability [15], [16], [17], [9]. In [15] the authors state -

> While it is useful to define such preprocessors, simply possessing a knob that can adjust computational complexity does not constitute a solution to the problem of fast spectral cluster- ing. What is needed is an explicit connection between the amount of data reduction that is achieved by a preprocessor and the sub- sequent effect on the clustering.

In this article, as a by product of calculating the $\Gamma-$limit for spectral clustering, we have a preprocessor which can be shown to be consistent with the results of spectral clustering.
(4) We propose an alternate method to handle the 'small

clusters' problem with the MST based clustering.

(A summary of the results - both speed and accuracy for image segmentation will be shown in the introduction.)

II. Background

In this section we provide the background for various concepts from spectral clustering, MST based clustering and $\Gamma-$limit, required for the rest of the article. In the later sections we show that the $\Gamma-$limit of spectral clustering is infact closely related to MST based clustering.

Let $X = \{x_i\}$ represents the dataset, where x_i denotes each data entry. Let $x_i \in \mathbb{R}^n$, where n 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 entries, each entry 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 choices for constructing the edge set and the similarity depending on the domain of application [8]. The choices made will be made explicit when required.

Let D be 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 then defined by

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

We know that the Laplacian is a symmetric positive-semi definite matrix, and hence has real eigenvalues greater than or equal to 0, 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

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

In this article we also work with *exponentiated graphs* - weighted graphs whose weights are raised to a power p. $W^{(p)}$ denotes the weight matrix of the exponentiated graph, i.e $W^{(p)}_{ij} = w^p_{ij}$. Let $D^{(p)}$ denote the matrix as constructed in equation (1) with weights w^p_{ij}. Let $L^{(p)} = D^{(p)} - W^{(p)}$.

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

A. Spectral Clustering

Spectral clustering methods work by projecting the data onto a subspace, such that the traditional methods for clustering work well on this projected dataset. There are 3 steps which form the spectral clustering methods:

1) Given a set of points $\{x_i\}$ (dataset), construct a graph with each point as a vertex. The edge set and the weights can be taken in a number of ways, as we shall discuss shortly. Let $\mathcal{G}$ denote the graph obtained.
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 one would like to obtain. let K be the matrix such that the i^{th} column of K is the i^{th} eigenvector e_{i-1}.
3) Using rows of the matrix K as new representation of the points x_i, use traditional clustering methods such as k-means to obtain the final clusters.

The laplacian in (2) is known as an unnormalized laplacian. Some consider the normalized laplacians, L_1, L_2 as well [2], [18].

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

It is not completely understood why spectral clustering methods work. However, there exists several results which provide insight into why these methods work.

Proposition 1. *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 $\{\mathbf{1}_{A_1}, \mathbf{1}_{A_2}, \cdots, \mathbf{1}_{A_k}\}$.*

The proposition 1 states that, if indeed the graph has k connected components, considering the first k eigenvectors would result in identifying the right subspace to project the data on, such that step 3 of the spectral clustering method above would work. Also, one can also show rigorously that, the eigenspaces of a matrix and its perturbation are 'close' by. These two observations allow us to infer that the spectral clustering methods would work in most cases [8].

1) 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 dissimilar 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. To solve this, it was proposed to use a slight modification of the above cost function. *Ratio cut* [8] is given by

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

where $|A_i|$ 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 can be shown that minimizing the $ratiocut(.,.)$ for k clusters is approximately equivalent to solving the optimization problem (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 this optimization problem is obtained by considering the first k eigenvectors of L as columns of H [19].

Another modification of the $cut()$ cost function, proposed in [2] is the *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 [19].

The first k eigenvectors of the respective laplacians only gives *a* solution to the optimization problems (7) and (9). In this article we characterize *all* the solutions to the optimization problems (7) and (9) in theorem 2 and is one of the contributions of this article.

B. Γ-convergence

Let $\min\{F_m(x) : x \in X\}$ be a family of minimum problems. The question of interest is the limiting behaviour 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. The advantages of such a substitution are - 1) This gives an approximate solution to the family of minimizers 2) Dependence on a parameter is nullified 3) Usually this results in a new method which would present a different model which was previously modeled with F_m. This is the problem which the theory of $\Gamma-$convergence addresses [13]. This method was recently used in [11] to calculate the $\Gamma-$limit of the cost function in [12].

Let $1 \geq \lambda_0 > \lambda_1 > \cdots > \lambda_k \geq 0$. Consider the cost function

$$Q^p(x) = \sum_{i=0}^{k}\lambda_i^p Q_i(x) \tag{10}$$

Our main interest is in calculating 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. In [14] a generic algorithm was proposed to calculate the limit of minimizers of (10). Assume that we are interested in finding solutions in a compact set C. Define

$$M_0 = \arg\min_{x \in C} Q_0(x) \tag{11}$$

i.e, M_0 is the set of minimizers for Q_0. Now recursively define,

$$M_i = \arg\min_{x \in M_{i-1}} Q_i(x) \tag{12}$$

Theorem 1. *Let X^* be the union of the sets of minimizers of Q_p (as defined in* (10)*) for all p. Then every limit point of X^* belongs to the set M_k.*

Refer [14] for the proof of the theorem 1. Intuitively, one can interpret the above theorem 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 an algorithm to calculate the $\Gamma-$limit [14].

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 the $\Gamma-$limit belongs to the output of algorithm 1. However, the converse is not true in general - all the 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.)

Proposition 2. *Let x^* be a $\Gamma-$limit for Q^p. Let $\overline{x}$ be a solution obtained from algorithm 1. Then we have that, for all p*

$$Q^p(x^*) = Q^p(\overline{x}) \tag{13}$$

The proof of the proposition 2 is straightforward. Thus, although algorithm 1 does not calculate the strict $\Gamma-$limit, we have that it calculates equivalent solutions as far as the cost function is concerned.

In this article, our main focus is to calculate the $\Gamma-$limit of the spectral clustering problem, In particular we are interested in calculating the limit of the minimizers of (7) for exponentiated graphs, $\mathcal{G}^p = (V, E, W^{(p)})$ as $p \to \infty$.

C. Maximum Spanning Tree Clustering

Let $\mathcal{G} = (V, E, W)$ be a connected similarity graph. A *spanning tree* 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 weighs of the edges in the tree. A spanning tree with the maximum weight is known as the maximum spanning tree (MST). There are several clustering methods based on MST [20]. Although MST based clusters is not so prevalent in clustering general data, for the image segmentation slight variations of MST based clustering was shown to be very useful, thanks to its equivalence to watersheds [21], [22]. A generic MST based clustering algorithm is

1. Given a similarity graph $\mathcal{G} = (V, E, W)$, construct a MST, T, and order the edges of T according to the weights.
2. If the required number of clusters is equal to n_c, add edges starting from highest weight until we reach the number of components n_c. 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. MST based clustering is prone to noise and outliers.

In practice, MST clustering gives small clusters and hence is not used widely. Also, the arbitrariness in breaking the ties gives non uniform results. Here we propose a different method in algorithm 2 to handle the small clusters problem. To our knowledge this has not been used before.

Algorithm 2 Minimum Spanning Tree Clustering

Input: A similarity graph, $\mathcal{G} = (V, E, W)$. Assume there are k distinct weights $0 < w_1 < w_2 < \cdots < w_k < 1$. Number of clusters, m. The edge set E is ordered, with ties breaking arbitrarily. $threshComp$ parameter indicating the minimum required size of the component.

Output: Cluster labels

```
set i = k
while number of components in G_{≥w_i} whose size is greater than threshComp is greater than m do
    i = i − 1
end while
Let Ĝ = G_{≥w_i}
for each edge e in E whose weight is w_i do
    Add edge e to Ĝ.
    if Number of connected components in Ĝ is equal to m then
        Break.
    end if
end for
Give a unique label to each of the m components.
Add all the points which are unlabelled to a queue Q
while Q is not empty do
    Take a point q ∈ Q.
    if no neighbor of q is labelled then
        continue
    else
        assign the most dominant label in the neighborhood of q to q
    end if
end while
```

Heuristically, algorithm 2 makes sure that the major components are not merged and initially ignores the small clusters. Then each point of the small cluster is then assigned a label according to the neighborhood. Figure 1 shows the results obtained with algorithm 2.

(a) (b) (c)

Fig. 1. MST based clustering. (a) Original Image. (b) and (c) 2 clusters obtained using algorithm 2.

III. Gamma limit of spectral clustering

Consider the optimization problem in (7). In the case of exponentiated graphs, the optimization problem is restated as

$$\begin{aligned} \underset{H \in \mathbb{R}^{n\times k}}{\text{minimize}} \quad & Tr(H^t L^{(p)} H) \\ \text{subject to} \quad & H^t H = I \end{aligned} \tag{14}$$

In this section, we calculate the $\Gamma-$limit (limit of minimizers) of the optimization problem in (14) as $p \to \infty$. The starting point of calculation is the generic algorithm 1.

We need some more notation. Assume $0 < w_1 < w_2 < \cdots < w_k < 1$ indicate the k distinct weights any edge can take. Denote by $\mathcal{G}_i$ the graph with the vertex set same as before, but the edge set consisting of only those edges whose weights are equal to w_i. Accordingly, one can define the laplacian for this graph L_k. Then we have,

$$Tr(H^t L H) = \sum_{i=1}^{k} w_i Tr(H^t L_i H) \tag{15}$$

For the exponentiated graphs,

$$Tr(H^t L^{(p)} H) = \sum_{i=1}^{k} w_i^p Tr(H^t L_i H) \tag{16}$$

Note that the above equation is in the form of (10) 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 $\mathbb{R}^n$. Note that 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. This matrix is not unique. Also, let $\mathcal{H}(M)$ denote the set of all matrices whose column space is equivalent to some subspace in a set of subspaces, M. Let $\mathcal{M}(H)$ denote the set of all column spaces of matrices in a set of matrices H. The following relations hold true.

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

where M' and H' are some sets of subspaces and matrices respectively. With this notation, the generic algorithm 1 in the case of ratio cut optimization would be the one in algorithm 3. 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 inturn characterize the M_i and $\mathcal{H}(M_i)$ at every stage i of the algorithm 3.

A. 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 $\lambda_{(m)}$. Here m is the number of clusters required. Let l be the total number of eigenvectors thus obtained. Let l_2 be the number

Algorithm 3 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

$$\begin{array}{ll} \underset{H\in\mathbb{R}^{n\times m}}{\text{minimize}} & Tr(H^t L_i H) \\ \text{subject to} & H^t H = \mathbf{I} \\ & H \in \mathcal{H}(M_i) \end{array} \tag{17}$$

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**

of eigenvectors whose eigenvalue is equal to $\lambda_{(m)}$, and l_1 be the number of remaining vectors. Let K be the matrix

$$K = \begin{bmatrix} I_{l_1\times l_1} & 0 \\ 0 & X_{l_2\times m-l_1} \end{bmatrix} \tag{18}$$

where $K^t K = I$. Also note that $l_1 + l_2 = l$. Let Y be an orthogonal matrix.

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

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

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

Since, the minimum value of $Tr(H^t LH)$ is equal to $\lambda_1 + \lambda_2 + \cdots + \lambda_m$ [19]. 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 (18) and Y is any orthogonal matrix. Hence all the solutions to (7) must be of the form AKY. □

Theorem 2 characterizes the sets $\mathcal{H}(M_i)$ and M_i at every stage. At 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 17 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*

$$\mathcal{H}(\mathcal{M}(A_k KY)) = \{ \textit{ matrices of the form } A_k KY\} \tag{19}$$

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. □

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

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

which is equivalent to solving the optimization problem

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

Now, since Y is orthogonal, we have $Tr(Y^t K^t A_k^t L_{k-1} A_k KY) = Tr(K^t A_k^t L_{k-1} A_k K)$. And, noting the special form of K, solving the above optimization problem is equivalent to solving the optimization problem

$$\begin{array}{ll} \underset{X}{\text{minimize}} & Tr(X^t A_k^t L_{k-1} A_k X) \\ \text{subject to} & X^t X = \mathbf{I} \end{array}$$

Observe that the matrix $A_k^t L_{k-1} A_k$ is symmetric positive semi-definite. And hence theorem 2 applies. Call $\hat{A}_{k-1}$ the matrix obtained by stacking the eigenvectors of $A_k^t L_{k-1} A_k$ whose eigenvalue is less than or equal to λ_{m-l_1}. Then all the solutions are of the form $\hat{A}_{k-1} KY$. Let

$$A_{k-1} = \begin{bmatrix} I_{l_1 \times l_1} & 0 \\ 0 & \hat{A}_{k-1} \end{bmatrix} \tag{20}$$

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. Proceeding as before, we get algorithm 4 to calculate the gamma limit.

Algorithm 4 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 $i = k$, $N_i = \mathbf{I}_n$, $l_1 = 0$, $l_2 = n$
2: $\{l_2$ indicates the number of eigenvectors at the end whose eigenvalue is the same.$\}$
3: Construct the scale graph $\mathcal{G}_i$ at level i, and laplacian L_i.
4: Construct matrix $C = [N_i^t L_i N_i]_{l_2, l_2}$ where $Z_{p,p}$ denotes the bottom right block of matrix Z whose dimension is $p \times p$.
5: Calculate the first eigenvectors of the generalized eigenvalue problem

$$Cx = \lambda x \tag{21}$$

such that the eigenvalue is less than or equal to $\lambda_{(m)}$. Let $\{a_1, a_2, \cdots, a_l\}$ be the eigenvectors calculated.
6: Let A be the matrix with the vectors $\{a_1, a_2, \cdots, a_l\}$ stacked as columns.
7: Construct $\hat{A}$ as

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

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

Intuitively, algorithm 4 stars with a trivial representation of the points $\mathbf{I}_n$. At each level of the algorithm, the representation is updated by considering the solutions of the optimization problem. One is iteratively reducing the column space to which the data can belong given that they have to solve the optimization problem 17.

Although algorithm 4 is implementable, note that at every stage one does a lot of matrix manipulations and calculates the eigenvectors. Calculation of eigenvectors is both error prone and computationally expensive.

IV. Efficient Implementation

In this section, we provide an efficient alternative of the algorithm 4. Recall that a connected component of a graph $\mathcal{G}$ is the maximal subgraph of $\mathcal{G}$ which is connected. This is obtained using the following proposition.

Proposition 3. *Let $\mathcal{G}$ be a similarity graph. At a given level j, let $\{C_1, C_2, \cdots C_m\}$ be the connected components of the graph $\mathcal{G}_{\geq w_j}$. 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 (17) *at level j is of the form CY where Y is any orthogonal matrix.*

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 (17) 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. □

Proposition 3 allows us to optimize the first steps of the algorithm 4 by considering the connected components instead of the eigenvectors. Intuitively this is due to proposition 1. We thus have algorithm 5. We refer to the solution of the algorithm 5 as the *Power Rcut*.

An interesting remark is that - the first stage of the algorithm simply involves computing the threshold of the graph and calculating the connected components. This is equivalent to the MST clustering as described in II-C. Thus, intuitively the gamma limit of spectral clustering is MST clustering, with one important difference - Power Rcut does not break ties arbitrarily like the MST based clustering. Rigorously, of course, the difference between MST based clustering and Power Rcut are slightly more involved. We shall analyze and compare the Power Rcut and MST based clustering in the next section.

Note that the first phase of the algorithm 5 (MST phase) takes time of the order $\mathcal{O}(|V| + |E|)$. The second phase of the algorithm calculates the eigenvectors and is of the order $\mathcal{O}(n^3)$ where n is the size of the matrix. But in many practical situations $n \sim \mathcal{O}(m) << |V|$, where m denotes the number of clusters. Hence the overall complexity for all practical purposes is $\mathcal{O}(|V| + |E|)$. Thus, intuitively, Power Rcut algorithm scales well for big data compared to the spectral clustering procedures.

V. Implementation Details

To summarize, we have shown that algorithm 5 calculates the gamma limit of spectral clustering. Note that we have assumed that there are only k distinct weights an edge can take where $k << |E|$. However, in reality, due to floating

Algorithm 5 Efficient algorithm to compute $\Gamma-$limit for ratio-cut.

Input: A weighted graph, $\mathcal{G}$, with distinct weights $w_1 < w_2 < \cdots < w_k$. Number of clusters, 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: Construct N by stacking the vectors $\mathbf{1}_{A_j}/\sqrt{|A_j|}$ in columns, where A_j is a connected component of $\mathcal{G}_{\geq w_i}$.
6: Set $l_1 = 0$ and $l_2 =$ number of connected components in $G_{\geq w_i}$
7: Consider the graph $\mathcal{G}_i$ and let L_i be the corresponding laplacian.
8: Set $C = [N^t L_k N]_{l_2, l_2}$
9: Calculate the first eigenvectors of eigenvalue problem whose eigenvalue is less than or equal to $\lambda_{(m)}$.

$$Cx = \lambda x$$

10: Let A be the matrix obtained by stacking the eigenvectors as columns.
11: Construct $\hat{A}$ as

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

12: Update l_1 and l_2.
13: $N = N \times \hat{A}$
14: Set $i = i - 1$
15: **if** i = 0 **or** number of columns of N is equal to m **then**
16: **return** N
17: **else**
18: Goto Step (7)
19: **end if**

point precision and other numerical errors identifying the k distinct weights is a challenge. Thus we resort to *'bucketing' of weights* to identify the distinct weights. In this section we discuss two methods to bucket the edge weights and their theoretical justification.

ϵ **bucketing :** Note that a result of bucketing of weights should club the weights which are 'close' by. Assume that the reason one obtains a lot of distinct weights is precision error. Then, change the weights according to

$$w_{ij} \longrightarrow \epsilon \times \text{round}(w_{ij}/\epsilon) \tag{22}$$

Here ϵ is the precision controlling parameter. Observe that for $\epsilon = 0.01$, (22) takes the first two precision digits after the decimal point.

K-means bucketing : The problem of bucketing the weights can be rephrased as - Combine the weights into buckets so that weights within a bucket are 'alike'. This is equivalent to clustering problem and hence one can use any clustering method to cluster the weights. In this article we chose $K - means$ for its efficiency and simplicity. In this case each weight is represented by the mean of the cluster to which the weight belongs to.

So, how does bucketing the weights change the algorithm? Let the bucketed weight of w_{ij} be denoted by $\hat{w}_{ij}$. Then,

$$w_{ij} = \frac{w_{ij}}{\hat{w}_{ij}} \times \hat{w}_{ij} \tag{23}$$

Recall that we have calculated the limit of minimizers as $p \to \infty$ in (14). We instead, define $W^{(p)}$ by

$$W^{(p)} = \hat{W}^{(p)} \odot W \tag{24}$$

where $\odot$ indicates the Hadamard product (entry wise product), and $\hat{W}^{(p)} = [\hat{w}_{ij}^p]$. It is easy to the see that the rest of the arguments follow accordingly, with one key difference - the laplacian L_i is no longer a $0-1$ matrix in step 7 of algorithm 5. This is identical to the extension of Power-watershed framework proposed in [14].

Another place where algorithm 5 suffers in practice is in steps 9-12. As observed before, solutions to eigenvalue problems are error prone and cascading the solutions to the eigenvalue problems is not numerically optimal. In practice it has been observed that non-zero eigenvalues rarely repeat for the eigenvalue problem in solutions to (17). Apart from that, the number of components after adding the edges from the penultimate bucket is usually found to be $>> m$ (required number of clusters). Hence taking these assumptions into account, one can replace algorithm 5 with algorithm 6.

Algorithm 6 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: {according to the assumption we have number of connected components in graph $\mathcal{G}_{\geq w_2}$ is greater than m.}
2: Let $\{C_i\}, i \in \{1, 2, \cdots, n_c\}$ be the connected components in $\mathcal{G}_{\geq w_2}$.
3: Let I_{C_i} be the vector

$$I_{C_i}(x) = \begin{cases} 1/\sqrt{|C_i|} & \text{if } x \in C_i \\ 0 & \text{othwerwise} \end{cases} \tag{25}$$

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

Observe that algorithm 6 has a simple interpretation. We preprocess the graph with MST based clustering to reduce the number of vertices. And then use spectral methods on the resulting contracted graph to get the clusters.

REFERENCES

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

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

[3] Makoto Iwayama and Takenobu 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.

[4] Varun Chandola, Arindam Banerjee, and Vipin Kumar, "Anomaly detection: A survey," *ACM computing surveys (CSUR)*, vol. 41, no. 3, pp. 15, 2009.

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

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

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

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

[9] Yangqiu Song, Wen-Yen Chen, Hongjie Bai, Chih-Jen Lin, and Edward Chang, "Parallel spectral clustering," *Machine Learning and Knowledge Discovery in Databases*, pp. 374–389, 2008.

[10] Jordi Pont-Tuset, Pablo Arbelaez, Jonathan T Barron, Ferran Marques, and Jitendra 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.

[11] Camille Couprie, Leo Grady, Laurent Najman, and Hugues Talbot, "Power watershed: A unifying graph-based optimization framework," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 33, no. 7, pp. 1384–1399, 2011.

[12] Ali Kemal Sinop and Leo 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.

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

[14] Laurent Najman, "Extending the powerwatershed framework thanks to Γ-convergence," Tech. Rep., Université Paris-Est, LIGM, ESIEE Paris, Jan. 2017.

[15] Donghui Yan, Ling Huang, and Michael 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.

[16] Jie Chen, Haw-ren Fang, and Yousef 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.

[17] Charless Fowlkes, Serge Belongie, Fan Chung, and Jitendra Malik, "Spectral grouping using the nystrom method," *IEEE transactions on pattern analysis and machine intelligence*, vol. 26, no. 2, pp. 214–225, 2004.

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

[19] Helmut Lütkepohl, *Handbook of Matrices*, Wiley, 1 edition, 1997.

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

[21] Jean Cousty, Gilles Bertrand, Laurent Najman, and Michel 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.

[22] Jean Cousty, Gilles Bertrand, Laurent Najman, and Michel Couprie, "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.