1 Paper Title

On Spectral Clustering : Analysis and an algorithm

2 Summary

The problem discussed in this paper [3] is that of efficient clustering. Parametric density estimators usually rely on a myriad of assumptions to give good results, while also suffering due to the dependence of EM algorithm on the initialization. The method the authors concern themselves with is an alternative known as "Spectral Clustering". While many papers have already been written on the method, the major contribution comes as part of proposing an algorithm that uses the k-eigenvectors of the Laplacian matrix of the graph and gives a better performance.

3 Detailed Analysis

Let's discuss the steps of the algorithm and understand why it works.
Given set of points $S = s_1, s_2, s_3, ..., s_n \in R^l$ and we want to cluster these. Let's assume we want k clusters.

1. The first step is to build a similarity/affinity matrix $A \in R^{nxn}$ using Gaussian density for similarity: $A_{ii} = exp(-||s_i - s_j||^2/2\sigma^2)$ if $i \neq j$ and $A_{ij} = 0$ if $i = j$
2. The next step is to define a D matrix with $D_{ii} = \Sigma_j A_{ij}$ and then construct the Laplacian from it as $L = D^{\frac{-1}{2}} A D^{\frac{-1}{2}}$
3. Perform eigendecomposition on L, and get the k largest eigenvectors. Normalize the eigenvectors and get the normalized matrix $Y \in R^{nxk}$
4. Apply K-means to Y (n points with k dimensions each) with k clusters and finally assign the original points to clusters

A natural question is - Why this particular algorithm works? The authors state some assumptions which allows them to prove the effectiveness of the algorithm.
We first assume that all the points in different clusters were infinitely apart and each cluster is connected. Then the rows of Y (the n points calculated) give k mutually orthogonal points $r_1, .., r_k$ .i.e.

$$y_j^{(i)} = r_i \tag{1}$$

, $\forall i = 1, .., k, j = 1, .., n$ The largest eigenvalue of L which is 1, with multiplicity equal to k would be used to estimate the number of clusters. These clusters depict the exact clustering of the original data (with the assumption in mind), and hence show the algorithm works in this case.
For the more general case, if the assumptions of clusters being infinitely apart were removed, the matrix A's non-diagonal elements become non-zero. The first assumption is using matrix perturbation theory, the authors formalize a relationship between the eigengap and the stability of the eigenvectors. Turns out, if the stability of the eigenvectors is not to be compromised, the assumption to be considered is that $1 - \lambda_2^{(i)} \geq \delta$, where $\delta \geq 0$, and 1 is the largest eigenvalue such that the equation signifies the eigengap. This heuristic suggests that the number of clusters k should be the value that maximizes the eigengap δ[4].
This can be understood intuitively as we want vectors that cluster around an eigenvector to be tight, and to be distant from one another. To formalize this, the authors use "Cheeger constant" $h(S_i)$ for each cluster $i \in k$ to be

$$\frac{h(S_i)^2}{2} \geq \delta \tag{2}$$

The second assumption has got to do with "connectedness" of points to points in its own cluster as well as points in other clusters. The paper states that

$$\Sigma_{j \in S_{i_1}} \Sigma_{k \in S_{i_2}} \frac{A_{jk}^2}{d_j d_k} \leq \epsilon_1 \tag{3}$$

where, $i_1, i_2 \in 1, .., k$, $i_1 \neq i_2$, ϵ_1 is a fixed constant, A_{jk} is the j,k entry of Similarity matrix and d_j is the j entry of the D matrix.
Essentially, A_{jk} measures how "connected" each point in cluster S_1 is to points in cluster S_2, and d_j measures how "connected" points in cluster 1 are to each other.
The next assumption is based on our previous assumption and states that the ratio

$$\frac{(\Sigma_{k:k\notin S_i} A_{jk})}{(\Sigma_{k:k\in S_i} A_{jk})} \tag{4}$$

should be small ensuring that points within a cluster are more connected and points among clusters are less connected.
The final assumption made by the authors is again linked with the concept of "connectedness" and asserts that in a cluster, all points be more or less similarly connected to the other points in the same cluster giving the equation

$$d_j^{(i)} \geq (\Sigma_{k=1}^{n_i} d_k^{(i)})/(Cn_i) \tag{5}$$

for constant $C \geq 0$
If all assumptions stated above in equations (2), (3), (4) and (5) are true, $\epsilon = \sqrt{k(k-1)\epsilon_1 + k\epsilon_2^2}$, and $\delta \geq (2+\sqrt{2})\epsilon$, then Y's rows can be approximated by:

$$\frac{1}{n}\Sigma_{i=1}^{k}\Sigma_{j=1}^{n_i}||y_j^(i)) - r_i||_2^2 \leq 4C(4+2\sqrt{k})^2\frac{\epsilon^2}{\delta-\sqrt{2}\epsilon} \tag{6}$$

where $r_1, .., r_k$ are k mutually orthogonal vectors. This again like Eqn (1) means that the rows of Y give us orthogonal vectors around which all points form clusters.

4 Your critiques

4.1 Pros and cons

This algorithm, despite its many assumptions, is simple to implement and gives better results as compared to algorithms using normalized cuts[2], recursive bipartitioning methods, and algorithms that use singular vectors directly and hence find a linear subspace[1]. In contrast, this paper uses Gaussian kernel to find similarity and finds the Lapacian of the A matrix. It gives good experimental results in less number of iterations and that is a major contribution of this paper. But the authors haven't stated anything for when the assumptions don't hold. If the assumptions that the r_i vectors are orthogonal doesn't hold, the Kmeans becomes cumbersome again (because now the initialization of clusters being orthogonal can't be done), and the algorithm is rendered inefficient like its predecessors. However, the algorithm is able to find clusters when regions are non-convex. This can be attributed to the fact that orthogonal vectors can still be found in such a case, and the similarity measure is rightly selected as well. Yet, the authors haven't talked about enough experiments in such a case to reliably conclude on this.

5 Closing Remarks

5.1 What you have learned from reading this paper ?

The major revelation that I had while reading this paper was how small tweaks to an algorithm can result in such different results. There were good spectral clustering algorithms already, but tweaks like using Gaussian similarity, using Laplacian matrix resulted in much better results. Also, the fact that intriducing non-linearity can boost the properties of an algorithm is a handy concept which I learnt.

References

[1] KANNAN, R., VEMPALA, S., AND VETTA, A. On clusterings: Good, bad and spectral. *Journal of the ACM (JACM) 51*, 3 (2004), 497–515.

[2] MEILA, M., AND SHI, J. Learning segmentation by random walks. In *Advances in neural information processing systems* (2001), pp. 873–879.

[3] NG, A. Y., JORDAN, M. I., AND WEISS, Y. On spectral clustering: Analysis and an algorithm. In *Advances in neural information processing systems* (2002), pp. 849–856.

[4] WANG, S., AND ROHE, K. Don't mind the (eigen) gap.