Handbook of Cluster Analysis

C. Hennig, M. Meila, F. Murtagh, R. Rocci (eds.)

February 28, 2015

Contents

Chapter 1

Spectral clustering

Marina Meila

Department of Statistics, University of Washington

Abstract

Spectral clustering is a family of methods to find K clusters using the eigenvectors of a matrix. Typically, this matrix is derived from a set of pairwise similarities S_{ij} between the points to be clustered. This task is called similarity based clustering, graph clustering, or clustering of diadic data.

One remarkable advantage of spectral clustering is its ability to cluster "points" which are not necessarily vectors, and to use for this a"similarity", which is less restrictive than a distance. A second advantage of spectral clustering is its flexibility; it can find clusters of arbitrary shapes, under realistic separations.

This chapter introduces the similarity based clustering paradigm, describes the algorithms used, and sets the foundations for understanding these algorithms. Practical aspects, such as obtaining the similarities are also discussed.

1.1 Similarity based clustering. Definitions and criteria

1.1.1 What is similarity based clustering?

Clusters when the data represent similarities between pairs of points is called *similarity based clustering*. A typical example of similarity based clustering is community detection in social networks [47] (see also Chapter **??**), where the observations are individual links between people, which may be due to friendship, shared interests, work relationships. The "strength" of a link can be the frequency of interactions, e.g. communications by e-mail, phone or other social media, co-authorships or citations.

In this clustering paradigm, the points to be clustered are not assumed to be part of a vector space. Their attributes (or features) are incorporated into a single dimension, the link strength, or *similarity*, which takes a numerical value S_{ij} for each pair of points i, j. Hence, the natural representation for this problem is by means of the *similarity matrix* $\mathbf{S} = [S_{ij}]_{i,j=1}^{n}$. The similarities are symmetric ($S_{ij} = S_{ji}$), and non-negative ($S_{ij} \geq 0$).

Less obvious domains where similarity based clustering is used include image segmentation, where the points to be clustered are pixels in an image, and text analysis, where words appearing in the same context are considered similar.

The goal of similarity based clustering is to find the global clustering of the data set that emerges from the pairwise interactions of its points. Namely, we want to put points that are similar to each other in the same cluster, dissimilar points in different clusters.

1.1.2 Similarity based clustering and cuts in graphs

It is useful to cast similarity based clustering in the language of graph theory. Let the points to be clustered $V = \{1, \ldots n\}$ be the nodes of a graph $\mathcal{G}$, and the graph edges be represented by the pairs i, j with $S_{ij} > 0$. The similarity itself is the weight of edge ij.

$$\mathcal{G} \;=\; (V, E), \quad E = \{(i,j),\, S_{ij} > 0\} \subseteq V \times V \tag{1.1}$$

Thus, $\mathcal{G}$ is an *undirected* and *weighted* graph. A partition of the nodes of a graph into K clusters is known as a (K-way) *graph cut*, therefore similarity based clustering can be viewed as finding a cut in the graph $\mathcal{G}$. The following definitions will be helpful. We denote

$$d_i \;=\; \sum_{j \in V} S_{ij} \tag{1.2}$$

the *degree* of node $i \in V$. The volume of V is Vol $V = \sum_{i \in V} d_i$. Similarly, we define the volume of cluster $C \subseteq V$ by

$$d_C = \sum_{i \in C} d_i.$$

Note that the volume of a single node is d_i.

The value of the cut between subsets $C, C' \subseteq V$, $C \cap C' = \emptyset$, briefly called the *cut* of C, C' is the sum of the edge weigths that cross between C and C'.

$$Cut(C, C') \;=\; \sum_{i \in C} \sum_{j \in C'} S_{ij}$$

Now we define the K-way *Cut* and respectively*Normalized Cut* associated to a partition $\mathcal{C} = (C_1, \ldots C_K)$ of V as

$$Cut(\mathcal{C}) \;=\; \frac{1}{2} \sum_{k=1}^{K} \sum_{k' \neq k} Cut(C_k, V \setminus C_k) \tag{1.3}$$

$$NCut(\mathcal{C}) \;=\; \sum_{k=1}^{K} \frac{Cut(C_k, V \setminus C_k)}{d_{C_k}}. \tag{1.4}$$

In particular, for $K = 2$,

$$NCut(C, C') \;=\; Cut(C, C') \left(\frac{1}{d_C} + \frac{1}{d_{C'}} \right)$$

Intuitively, a small $Cut(\mathcal{C})$ is indicative of a "good" clustering, as most of the removed edges must have zero or low similarity S_{ij}. For $K = 2$, $\text{argmin}_{|\mathcal{C}|=2} Cut(\mathcal{C})$ can be found tractably by the MinCut/MaxFlow algorithm [35]. For $K \geq 3$, minimizing the cut is

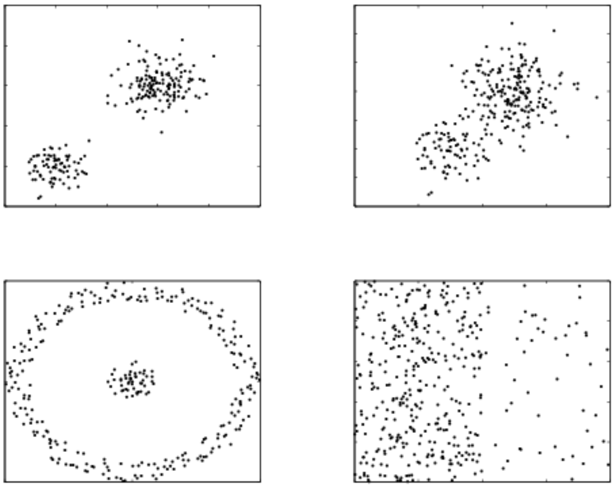


Figure 1.1: Four cases in which the minimum *NCut* partition agrees with human intuition.

NP-hard, in practice one applies the MinCut/MaxFlow recursively to obtain K-cuts of low value. Unfortunately, like the better known Single Linkage criterion, the *Cut* criterion is very sensitive to outliers; on most realistic dataset, the smallest cut will be between an outlier and the rest of the data. Consequently, clustering by minimizing *Cut* is found empirically to produce very *imbalanced* partitions[1].

This prompted [38] to introduce the *NCut* (which they called *balanced cut*). A partition can have small *NCut* only if it has both a small cut value and if all its cluster have sufficiently large volumes d_C. As Figure 1.1 shows, *NCut* is a very flexible criterion, capturing our intuitive notion of clusters in a variety of situations.

1.1.3 The Laplacian and other matrices of spectral clustering

In addition to the similarity matrix $\mathbf{S}$, a number of other matrices derived from it matrices play a central role in spectral clustering.

One such matrix is $\mathbf{P}$, the *random walk* matrix of $\mathcal{G}$, sometimes called the *random walk*

[1] An interesting *randomizing and averaging* algorithm using MinCut/MaxFlow was proposed by [18].

Table 1.1: The relevant matrices in spectral clustering

Matrix	name	dim	definition	properties
$\mathbf{S}$	similarity matrix	$n \times n$		$S_{ij} = S_{ji} \geq 0$
$\mathbf{D}$	degree matrix	$n \times n$	$\mathbf{D} = \mathrm{diag}(d_1, \ldots d_n)$	$D_{ii} = d_i > 0$, $D_{ij} = 0$, $j \neq$
$\mathbf{P}$	random walk matrix	$n \times n$	$\mathbf{P} = \mathbf{D}^{-1}\mathbf{S}$	$P_{ij} \geq 0$, $\sum_{j=1}^{n} P_{ij} = 1$
$\mathbf{L}$	Laplacian matrix	$n \times n$	$\mathbf{L} = \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{S}\mathbf{D}^{-1/2}$	$L_{ij} = L_{ji}$, $\mathbf{L} \succeq 0$
$\hat{\mathbf{P}}$	transition matrix btw. clusters	$K \times K$	$\hat{P}_{kl} = \sum_{i \in C_k} \sum_{j \in C_l} S_{ij}/d_{C_k}$	

Laplacian of $\mathcal{G}$. $\mathbf{P}$ is obtained by normalizing the rows of $\mathbf{S}$ to sum to 1.

$$\mathbf{P} \;=\; \mathbf{D}^{-1}\mathbf{S} \tag{1.5}$$

with $\mathbf{D}$ being the diagonal matrix of the node degrees

$$\mathbf{D} \;=\; \mathrm{diag}(d_1, \ldots, d_n) \tag{1.6}$$

Thus, $\mathbf{P}$ is a stochastic matrix, satisfying $P_{ij} \geq 0$, $\sum_{j=1}^{n} P_{ij} = 1$. Another matrix of interest is $\mathbf{L}$, the *Normalized Laplacian* [12] of $\mathcal{G}$, which we will call for brevity the *Laplacian*.

$$\mathbf{L} \;=\; \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{S}\mathbf{D}^{-1/2} \tag{1.7}$$

where $\mathbf{I}$ is the unit matrix.

Proposition 1 (Relationship between $\mathbf{L}$ and $\mathbf{P}$)**.** *Denote by* $1 = \lambda_1 \geq \lambda_2 \geq \;\ldots\; \lambda_n \geq -1$ *the eigenvalues of* $\mathbf{P}$ *and by* $\mathbf{v}^1, \ldots \mathbf{v}^n$ *the corresponding eigenvectors. Denote by* $\mu_1 \leq \mu_2 \leq \ldots\, \mu_n$ *the eigenvalues of* $\mathbf{L}$ *and by* $\mathbf{u}^1, \ldots \mathbf{u}^n$ *the corresponding eigenvectors. Then,*

1.

$$\mu_i \;=\; 1 - \lambda_i \quad \mathbf{u}^i \;=\; \mathbf{D}^{1/2}\mathbf{v}^i \quad \textit{for all } i = 1, \ldots n. \tag{1.8}$$

2. $\lambda_1 = 1$ *and* $\mu_1 = 0$

3. The multiplicity of $\lambda_1 = 1$ *(or, equivalently, of* $\mu_1 = 0$*) is* $K > 1$ *iff* $\mathbf{P}$ *(*$\mathbf{L}$*) is block diagonal with* K *blocks.*

This proposition has two consequences. Because $\lambda_j \leq 1$, it follows that $\mu_j \geq 0$; in other words, that $\mathbf{L}$ is positive semidefinite, with $\mu_1 = 0$. Moreover, Proposition 1 ensures that the eigenvalues of $\mathbf{P}$ are always real and its eigenvectors linearly independent.

1.1.4 Four bird's eye views of spectral clustering

We can approach the problem of similarity based clustering from multiple perspectives.

1. We can view each data point i as the row vector $S_{i:}$ in $\mathbb{R}^n$, and find a low dimensional embedding of these vectors. Once this embedding is found, one could proceed to cluster the data by e.g K-means algorithm, in the low-dimensional space. This view is captured by Algorithm 1.2 in 1.2.

2. We can view the data points as states of a Markov chain defined by $\mathbf{P}$. We group states by their *pattern of high-level connections* . This view is described in section 1.3.1.

3. We can view the data points as nodes of graph $\mathcal{G} = (V, E, S)$ as in Section 1.1.2. We can remove a set of edges with small total weight, so that none of the connected components of the remaining graph is too small, in other words we can cluster by minimizing the *NCut*. This view is further explored in Section 1.3.2.

4. We can view a cluster C as its $\{0, 1\}$-valued *indicator function* $\mathbf{x}_C$. We can find the partition whose K indicator functions are "smoothest" with respect to the graph $\mathcal{G}$, i.e. stay constant between nodes with high similarity. This view is described in Section 1.3.3.

As we shall see, the four paradigms above are equivalent, when the data is "well clustered", are are all implemented by the same algorithm, which we describe in the next section.

1.2 Spectral clustering algorithms

The workflow of a typical spectral clustering algorithm is shown in the top row of Figure 1.2.

The algorithm we recommend is based on [29, 30] and [34].

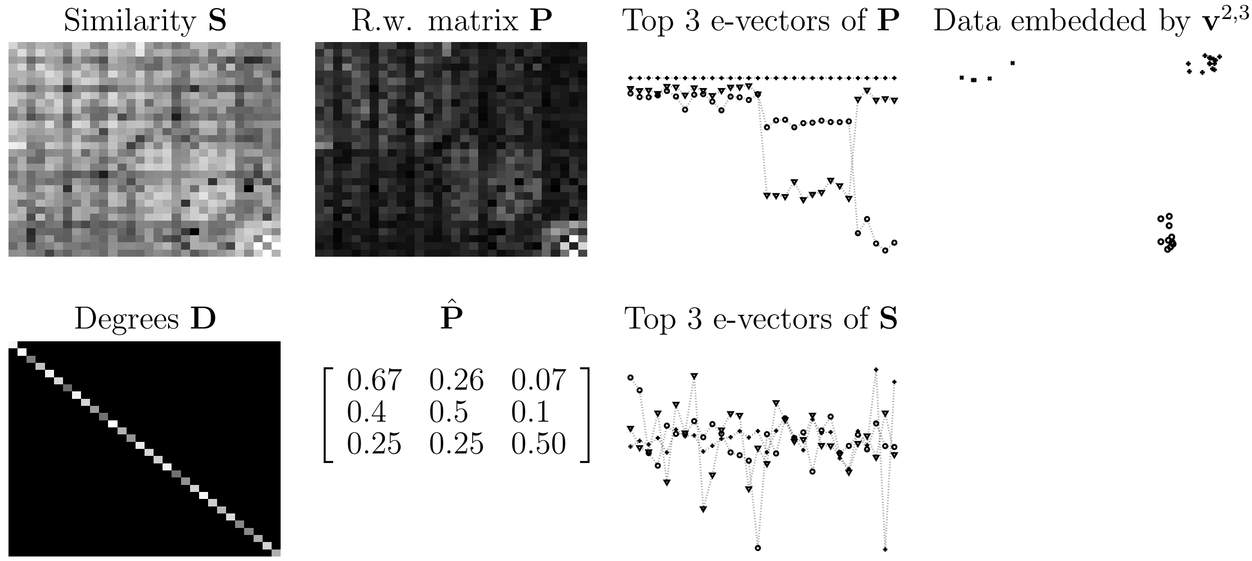
```
Algorithm SpectralClustering
Input Similarity matrix $\mathbf{S}$, number of clusters $K$
  1. Transform $\mathbf{S}$
     Calculate $d_i \leftarrow \sum_{j=1}^{n} S_{ij}$, $j = 1:n$ the node degrees.
     Form the transition matrix $\mathbf{P}$ with $P_{ij} \leftarrow S_{ij}/d_i$, for $i, j = 1:n$
  2. Eigendecomposition
     Compute the largest $K$ eigenvalues $\lambda_1 \geq \ldots \geq \lambda_K$ and eigenvectors $\mathbf{v}_1, \ldots \mathbf{v}_K$ of $\mathbf{P}$.
  3. Embed the data in $K$-th principal subspace
     Let $\mathbf{x}_i = [\, \mathbf{v}_{i2}\, \mathbf{v}_{i3}\, \ldots\, \mathbf{v}_{iK} \,] \in \mathbb{R}^{n \times (K-1)}$, for $i = 1, \ldots n$.
  4. Run the K-MEANS algorithm on the "data" $\mathbf{x}_{1:n}$
Output The clustering $\mathcal{C}$ obtained in step 4.
```

Note that in step 3 we discard the first eigenvector, as this is usually constant and is not informative of the clustering.

Some useful variations and improvements of SpectralClustering are:

- *Orthogonal initialization* [34] Find the K initial centroids $\bar{\mathbf{x}}_{1:K}$ of K-MEANS in step 4 by

```
Algorithm OrthogonalInitialization
  1. choose $\bar{\mathbf{x}}_1$ randomly from $\mathbf{x}_1, \ldots \mathbf{x}_n$
  2. for $k = 2, \ldots K$ set $\bar{\mathbf{x}}_k = \operatorname{argmin}_{\mathbf{x}_i} \max_{k' < k} |\cos(\bar{\mathbf{x}}_{k'}, \mathbf{x}_i)|$.
```



Similarity $\mathbf{S}$

R.w. matrix $\mathbf{P}$

Top 3 e-vectors of $\mathbf{P}$

Data embedded by $\mathbf{v}^{2,3}$

Degrees $\mathbf{D}$

$\hat{\mathbf{P}}$

$$\begin{bmatrix} 0.67 & 0.26 & 0.07 \\ 0.4 & 0.5 & 0.1 \\ 0.25 & 0.25 & 0.50 \end{bmatrix}$$

Top 3 e-vectors of $\mathbf{S}$

Figure 1.2: Spectral clustering of a synthetic data set with $n = 30$ points and $K = 3$ clusters of sizes 15, 10 and 5; the data are sorted so that points in the same cluster are consecutive. The **top row**, from left to right, displays the similarity matrix $\mathbf{S}$, the random walk matrix $\mathbf{P}$, the entries in the top 3 eigenvectors of $\mathbf{P}$, plotted vs. the index $i = 1, \ldots 30$, and finally, the embedding $\mathbf{x}_{1:n}$ of the data obtained from the eigenvectors. The similarity $\mathbf{S}$ is a perfect similarity matrix to which noise was added; hence in the second and third eigenvectors of $\mathbf{P}$ the corresponding to a cluster have approximately but not exactly the same value; the first eigenvector of $\mathbf{P}$ is proportional to $\mathbf{1}$ and hence has exactly equal entries for all i. Since $\mathbf{v}^{2,3}$ are almost piecewise-constant, in the embedding the points $\mathbf{x}_{1:n}$ are well clustered. The **bottom row** displays the node degrees on the diagonal of $\mathbf{D}$, the $\hat{P}_{kl}$ values of the transition probabilities between blocks, and the top 3 eigenvectors of $\mathbf{S}$. Note that this is not a case of nearly block diagonal $\mathbf{S}$: the probabilities of transitioning between clusters are significantly away from 0, and the minimum *NCut* is not small (its value is $1.33 = 3 - \operatorname{trace} \hat{\mathbf{P}}$). Yet the data is very “well clustered”, if one uses the eigenvectors of $\mathbf{P}$ for clustering. In contrast, the top 3 eigenvectors of the untransformed $\mathbf{S}$ are not informative (nor are the other eigenvectors of $\mathbf{S}$). The *Cut* corresponding to the clustering found by SpectralClustering is 140.3 (which represents 0.23 of the total $\operatorname{Vol} V = 614.5$); in contrast removing the point of smallest degree has *Cut* equal to 11.7.

This initialization is a variant of the FastestFirstTraversal algorithm [20]; FastestFirstTraversal is part of one the best EM andK-means initialization algorithms known to date [15, 10].

- *Rescaling* $\mathbf{x}_i$ *to have unit length* in step 3 was recommended by [34] and was found empirically to have good noise reduction effects.

- *Rescaling* $\mathbf{v}^{2:K}$ *by the eigenvalues (diffusion distance rescaling)* in step 2. When $\mathbf{P}$ is almost block diagonal, or close to perfect, this rescaling will have almost no effect. But in the noisier situations, it can put more weight on the first eigenvectors which are more robust to noise (see also Section 1.5). Moreover, [32] showed that setting $\mathbf{v}^k \leftarrow \lambda_k^{2t}\mathbf{v}^k$, with some $t > 1$, is related to the *diffusion distance*, a true metric on the nodes of a graph. The parameter t is a *smoothing parameter*, with larger t causing more smoothing.

- *Using* $\mathbf{S}$ *instead of* $\mathbf{P}$ in step 2 (and skipping the transformation in step 1). This algorithm variant can be shown to (approximately) minimize a criterion call *Ratio Cut* (*RCut*).

$$RCut(\mathcal{C}) = \sum_{k=1}^{K} \frac{Cut(C_k, V \setminus C_k)}{|C_k|} \tag{1.9}$$

  The *RCut* differs from the *NCut* only in the denominators, which are the cluster cardinalities, instead of the cluster volumes. The discussion in Sections 1.3.1,1.3.2 and 1.3.3 applies with only small changes to this variant of SpectralClustering, w.r.t. the *RCut* criterion. However, it can be shown that whenever $\mathbf{S}$ has piecewise constant eigenvectors (see Section 1.3.1) then $\mathbf{P}$ will have piecewise constant eigenvectors as well, but the converse is not true [45]. Hence, whenever this algorithm variant can find a good clustering, the original 1.2 can find it too. Moreover, the eigenvectors and values of $\mathbf{P}$ converge to well-defined limits when $n \to \infty$, whereas those of $\mathbf{S}$ may not.

The most significant variant of Algorithm 1.2 is its original recursive form [38] given below.

```
Algorithm TWO-WAY SPECTRAL CLUSTERING
Input Similarity matrix $\mathbf{S}$
  1. Transform $\mathbf{S}$
     Calculate $d_i = \sum_{j=1}^{n} S_{ij}$, $j = 1 : n$ the node degrees.
     Form the transition matrix $\mathbf{P}$ with $P_{ij} \leftarrow S_{ij}/d_i$ for $i, j = 1, \dots n$
  2. Compute the eigenvector $\mathbf{v}$ corresponding to the second largest eigenvalue $\lambda_2$ of $\mathbf{P}$
  3. Sort
     Let $\mathbf{v}^{sort} = [v_{i_1}\, v_{i_2}\, \dots\, v_{i_n}]$ be the entries of $\mathbf{v}$ sorted in increasing order and denote
     $C_j = \{i_1, i_2, \dots i_j\}$ for $j = 1, \dots n-1$.
  4. Cut
     For $j = 1, \dots n-1$ compute $NCut(C_j, V \setminus C_j)$ and find $j_0 = \operatorname{argmin}_j NCut(C_j, V \setminus C_j)$.
Output clustering $\mathcal{C} = \{C_{j_0}, V \setminus C_{j_0}\}$
```

TWO-WAY SPECTRAL CLUSTERING is called recursively on each of the two resulting clusters, if one wishes to obtain a clustering with $K > 2$ clusters.

Finally, an observation related to numerical implementation that is too important to omit. From Proposition 1, it follows that steps 1 and 2 of SPCALG can be implemented equivalently as

```
Algorithm STABLESPECTRALEMBEDDING
  1. $\tilde{L}_{ij} \leftarrow S_{ij}/\sqrt{d_i d_j}$ for $i, j = 1 : n$ (note that $\tilde{L} = I - L$)
  2. Compute the largest $K$ eigenvalues $\lambda_1 = 1 \geq \lambda_2 \geq \dots \geq \lambda_K$ and eigenvectors $\mathbf{u}^1, \dots \mathbf{u}^k$
     of $\tilde{\mathbf{L}}$ (these are the eigenvalues of $\mathbf{P}$ and the eigenvectors of $\mathbf{L}$).
     Rescale $\mathbf{v}_k \leftarrow \mathbf{D}^{-1/2}\mathbf{u}^k$ (obtain the eigenvectors of $\mathbf{P}$).
```

Eigenvector computations for symmetric matrices like $\tilde{\mathbf{L}}$ are much more stable numerically than for general matrices like $\mathbf{P}$. This modification guarantees that the eigenvalues will be real and the eigenvectors orthogonal.

1.3 Understanding the spectral clustering algorithms

1.3.1 Random walk/Markov chain view

Recall the stochastic matrix $\mathbf{P}$ defines a Markov chain (or random walk) on the nodes V. Remarkably, the stationary distribution π of this chain has the explicit and simple form[2]

$$\pi_i = \frac{d_i}{\operatorname{Vol} V} \quad \text{for } i \in V \tag{1.10}$$

Indeed, it is easy to verify that

$$[\pi_1 \ldots \pi_n]P = \frac{1}{\operatorname{Vol} V}\left[\sum_i d_i P_{i1} \ldots \sum_i d_i P_{in}\right] = \frac{1}{\operatorname{Vol} V}[\sum_{i=1}^n S_{i1} \ldots \sum_{i=1}^n S_{in} = [\pi_1 \ldots \pi_n] \tag{1.11}$$

If the Markov chain is ergodic, then π is the unique stationary distribution of $\mathbf{P}$, otherwise, uniqueness is not guaranteed, yet property 1.11 still holds.

Now let's consider the Algorithm 1.2 and ask when are the points $\mathbf{x}_i \in \mathbb{R}^K$ well clustered? Is there a case when the $\mathbf{x}_i$'s are identical for all the nodes i that belong to the same cluster k? If this happens we say that $\mathbf{S}$ (and $\mathbf{P}$) are *perfect*. In the perfect the case, the K-MEANS algorithm (or, by that matter, any clustering algorithm) will be guaranteed to find the same clustering.

Thus, to understand what is a "good" clustering from the point of view of spectral clustering, it is necessary to understand what the perfect case represents.

Definition 1. *If $\mathcal{C} = (C_1, \ldots C_K)$ is a partition of V, we say that a vector $\mathbf{x}$ is* piecewise constant *w.r.t $\mathcal{C}$ if for all pairs i, j in the same cluster C_k we have $x_i = x_j$.*

Proposition 2. *Lumpability Lemma [30] Let $\mathbf{P}$ be a matrix with rows and columns indexed*

[2] This is true for any *reversible* Markov chain.

by V that has independent eigenvectors. Let $\mathcal{C} = (C_1, C_2, \ldots C_k)$ be a partition of V. Then, $\mathbf{P}$ *has K eigenvectors that are piecewise constant w.r.t. $\mathcal{C}$ and correspond to non-zero eigenvalues if and only if the sums $P_{ik} = \sum_{j \in C_k} P_{ij}$ are constant for all $i \in C_l$ and all $k, l = 1, \ldots K$ and the matrix $\hat{\mathbf{P}} = [\hat{P}_{kl}]_{k,l=1,\ldots K}$ (with $\hat{P}_{kl} = \sum_{j \in C_k} P_{ij}$, $i \in C_l$) is non-singular. We say that (the Markov chain represented by)* $\mathbf{P}$ *is* lumpable *w.r.t $\mathcal{C}^*$.*

Corrolary 3. *If stochastic matrix* $\mathbf{P}$ *obtained in Step 1 is lumpable w.r.t $\mathcal{C}^*$ with piecewise constant eigenvectors $\mathbf{v}_1, \ldots \mathbf{v}_K$ corresponding to the K largest eigenvalues of* $\mathbf{P}$*, then Algorithm 1.2 will output $\mathcal{C}^*$.*

Corrolary 3 shows that spectral clustering will find clusterings for which points i, i' are in the same cluster k if they have the same probability $\hat{P}_{kl}$ of transitioning to cluster l, for all $k = l$ to K.

A well-known special case of lumpability is the case when the clusters are *completely separated*, i.e. when $S_{ij} = 0$ whenever i, j are in different clusters. Then, $\mathbf{S}$ and $\mathbf{P}$ are block diagonal with K blocks, each block representing a cluster. From Proposition 2 it follows that $\mathbf{P}$ has K eigenvalues equal to 1, and that $\hat{\mathbf{P}} = \mathbf{I}$. What can be guaranteed in the vicinity of this case has been intensely studied in the literature. In particular, [34] and later [5] give theoretical results showing that if $\mathbf{S}$ is nearly block diagonal, the clusters representing the blocks of $\mathbf{S}$ can be recovered by spectral clustering.

The Lumpability Lemma shows however that having an approximately block diagonal $\mathbf{S}$ is not necessary, and that spectral clustering algorithms will work in a much broader range of cases, namely as long as "the points in the same cluster behave approximately in the same way" in the sense of Proposition 2.

This interpretation relates spectral clustering to a remarkable fact about Markov chains. It is well-known that if one groups the states of a Markov chains in clusters $C_1, \ldots C_K$, a sequence of states $i_1, i_2, \ldots i_t$ implies a sequence of cluster labels $k_1, k_2, \ldots k_t \in \{1, \ldots K\}$. From the transition matrix $\mathbf{P}$ and the clustering $C_1, \ldots C_K$ one can calculate the transition matrix at the cluster level $Pr[C_k \to C_l | C_k] = \hat{P}_{kl}$, as well as the stationary distribution w.r.t the clusters by

$$\hat{P}_{kl} \;=\; \sum_{i \in C_k} \sum_{j \in C_l} S_{ij}/d_{C_k}, \quad \hat{\pi}_k \;=\; \frac{d_{C_k}}{\operatorname{Vol} V}, \quad k, l = 1, \dots K. \tag{1.12}$$

However, it can be easily shown that the chain $k_1, k_2, \dots k_t, \dots$ is in general not Markov; that is, $Pr[k_{t+1}|k_t, k_{t-1}] \neq Pr[k_{t+1}|k_t]$, or knowing past states *can* give information about future states even when the present state k_t is known. *Lumpability* in Markov chain terminology means that there exists a clustering $\mathcal{C}^*$ of the nodes in V so that the chain defined by $\hat{\mathbf{P}}$ is Markov. Proposition 2 shows that lumpability hold essentially iff $\mathbf{P}$ has piecewise-constant eigenvectors. Hence, spectral clustering Algorithm 1.2 finds *equivalence classes* of nodes (when they exist) so that all nodes in an equivalence class C_k contain the same information about the future.

The following proposition underscores the discussion about lumpability, showing that the eigenvectors of $\mathbf{P}$, when they are piecewise constant, are “stretched versions” of the eigenvectors of $\hat{\mathbf{P}}$.

Proposition 4. *Relationship between* $\mathbf{P}$ *and* $\hat{\mathbf{P}}$ *(Telescope Lemma) Assume that the conditions of Proposition 2 hold. Let* $\mathbf{v}^1, \dots \mathbf{v}^K \in \mathbb{R}^n$ *and* $1 = \lambda_1 \geq \lambda_2 \geq \; \dots \; \lambda_K$ *be the piecewise constant eigenvectors of* $\mathbf{P}$ *and their eigenvalues and* $1 = \hat{\lambda}_1 \geq \hat{\lambda}_2 \geq \; \dots \; \hat{\lambda}_K$ *and* $\hat{v}^1, \dots \hat{\mathbf{v}}^K \in \mathbb{R}^K$ *the eigenvalues and eigenvectors of* $\hat{\mathbf{P}}$*. Then*

$$\begin{aligned} \hat{\lambda}_k &= \lambda_k \quad \textit{and} & (1.13) \\ \hat{v}^k_l &= v^k_i \quad \text{for } l = 1, \dots K \quad \text{and } i \in C_l & (1.14) \end{aligned}$$

1.3.2 Spectral clustering as finding a small balanced cut in $\mathcal{G}$

We now explain the relationship between spectral clustering algorithms like 1.2 and minimizing the K-way normalized cut.

First, we show that the *NCut* defined in 1.4 can be rewritten in terms of probabilities $\hat{P}_{kl}$ of transitioning between clusters in the random walk defined by $\mathbf{P}$.

Proposition 5 (*NCut* as conditional probability of leaving a cluster)**.** *The K-way normalized cut associated to a partition $\mathcal{C} = (C_1, \dots C_K)$ of V is equal to*

$$NCut(\mathcal{C}) \;=\; \sum_{k=1}^{K} \left[1 - \frac{\sum_{i \in C_k} \pi_i \sum_{j \in C_k} P_{ij}}{\sum_{i \in C_k} \pi_i} \right] \;=\; \sum_{k=1}^{K} \left[1 - \hat{P}_{kk} \right] \;=\; K - \operatorname{trace} \hat{\mathbf{P}} \tag{1.15}$$

The denominators $\sum_{i \in C_k} \pi_i$ above represent $d_{C_k} / \operatorname{Vol} C_k = \pi_{C_k}$, the probability of being in cluster C_k under the stationary distribution π. Consequently each term of the sum represents the probability of leaving cluster C_k given that the Markov chain is in C_k, under the stationary distribution.

In the perfect case, from Proposition 4, $\hat{\lambda}_{1:K}$ are also the top K eigenvalues of $\mathbf{P}$, hence

$$NCut(\mathcal{C}^*) \;=\; K - \sum_{k=1}^{K} \lambda_k \tag{1.16}$$

Next, we show that the value $K - \sum_{k=1}^{K} \lambda_k$ is the lowest possible *NCut* value for any K-clustering $\mathcal{C}$ in any graph.

Proposition 6 (Multicut Lemma)**.** *Let* $\mathbf{S}$*,* $\mathbf{L}$*,* $\mathbf{P}$*,* $v^1, \dots v^K$ *and* $\lambda_1, \dots \lambda_K$ *be defined as before, and let* $\mathcal{C}$ *be a partition of* V *into* K *disjoint clusters. Then,*

$$\begin{aligned} NCut(\mathcal{C}) \;&\geq\; \min\{\operatorname{trace} \mathbf{Y}^T \mathbf{L} \mathbf{Y} \,|\, \mathbf{Y} \in \mathbb{R}^{n \times K},\, Y \textit{ has orthonormal columns}\} && (1.17) \\ &=\; K - (\lambda_1 + \lambda_2 + \dots + \lambda_K) && (1.18) \end{aligned}$$

The proof is both simple and informative so we will present it here. Consider an arbitrary partition $\mathcal{C} = (C_1, \dots C_K)$. Denote by $x^k \in \{0,1\}^n$ the indicator vector of cluster C_k for $k = 1, \dots K$.

We start with rewriting, again, the expression of *NCut* . From Proposition 5, noting that $\sum_{i \in C_k} d_i \;=\; \sum_{i \in V} (x_i^k)^2 d_i$ and

$$\sum_{i,j \in C_k} S_{ij} \;=\; \sum_{i,j \in V} S_{ij} x_i^k x_j^k \;=\; \sum_{i \in V} (x_i^k)^2 d_i - \sum_{ij \in E} S_{ij} (x_i^k - x_j^k)^2 \tag{1.19}$$

we obtain that

$$NCut(\mathcal{C}) \;=\; K - \sum_{k=1}^{K} \frac{\sum_{i,j \in C_k} S_{ij}}{\sum_{i \in C_k} d_i} \;=\; \sum_{k=1}^{K} \frac{\sum_{ij \in E} S_{ij}(x_i^k - x_j^k)^2}{\sum_{i \in V} (x_i^k)^2 d_i} \;=\; \sum_{k=1}^{K} R(x^k) \quad (1.20)$$

In the sums above, $i, j \in C_k$ means summation over the ordered pairs (i, j) while $ij \in E$ means summation over all "edges", i.e all unordered pairs (i, j) with $i \neq j$. Next, we substitute

$$\mathbf{y}^k \;=\; \mathbf{D}^{1/2}\mathbf{x}^k \quad (1.21)$$

obtaining

$$R(\mathbf{x}^k) \;=\; \frac{(\mathbf{y}^k)^T \mathbf{L} \mathbf{y}^k}{(\mathbf{y}^k)^T \mathbf{y}^k} \;=\; \tilde{R}(\mathbf{y}^k) \quad (1.22)$$

and

$$NCut(\mathcal{C}) \;=\; \sum_{k=1}^{K} \tilde{R}(\mathbf{y}^k) \quad (1.23)$$

The expression $\tilde{R}(\mathbf{y})$ represents the *Rayleigh quotient* for the symmetric matrix $\mathbf{L}$ [12] equation (1.13). Recall a classic Rayleigh-Ritz theorem in linear algebra [43], stating that the sum of K Rayleigh quotients depending on orthogonal vectors $\mathbf{y}^1 \ldots \mathbf{y}^K$ is minimized by the eigenvectors of $\mathbf{L}$ corresponding to its smallest K eigenvalues $\mu_1 \leq \mu_2 \leq \ldots \mu_K$. As $\mathbf{y}^k, \mathbf{y}^l$ defined by 1.21 are orthogonal, the expression 1.23 cannot be smaller than $\sum_{k=1}^{K} \tilde{R}(\mathbf{u}^k) = \sum_{k=1}^{K} \mu_k = K - \sum_{k=1}^{K} \lambda_k$, which completes the proof.

Hence, if $\mathbf{S}$ is perfect with respect to some K-clustering $\mathcal{C}^*$, then $\mathcal{C}^*$ is the minimum *NCut* clustering, and Algorithm 1.2 returns $\mathcal{C}^*$.

Recall that finding the clustering $\mathcal{C}^\dagger$ that minimizes *NCut* is NP-hard. Formulated in terms of $\mathbf{y}^{1:K}$ this problem is

$$\min_{\mathbf{y}^1, \ldots \mathbf{y}^K \in \mathbb{R}^n} \sum_{k=1}^{K} (\mathbf{y}^k)^T \mathbf{L} \mathbf{y}^k \quad \text{s.t.} \quad (\mathbf{y}^l)^T \mathbf{y}^k \;=\; \delta_{kl} \quad (1.24)$$

$$\text{there exist } \mathbf{x}^{1:K} \in \{0, 1\}^n \text{ so that 1.21 holds} \quad (1.25)$$

By dropping constraint 1.25, we obtain

$$\min_{\mathbf{y}^1,\ldots\mathbf{y}^K \in \mathbb{R}^n} \sum_{k=1}^{K} (\mathbf{y}^k)^T L \mathbf{y}^k \;\; \text{s.t.} \;(\mathbf{y}^l)^T \mathbf{y}^k \;=\; \delta_{kl} \tag{1.26}$$

whose solution is given by the eigenvectors $\mathbf{u}^1, \ldots, \mathbf{u}^K$ and smallest eigenvalues $\mu^1, \ldots \mu^K$ of $\mathbf{L}$. Applyiing 1.21 and Proposition 1 to $\mathbf{u}^{1:K}$ we see that the $\mathbf{x}^{1:K}$ correspondig to the solution of 1.26 are no other than the eigenvectors $\mathbf{v}_{1:K}$ of $\mathbf{P}$. Problem 1.26 can be formulated directly in the $\mathbf{x}$ variables as

$$\min_{\mathbf{x}^1,\ldots\mathbf{x}^K} \sum_{k=1}^{K} R(\mathbf{x}^k) \;\; \text{s.t.} \;\; \mathbf{x}^k \perp \mathbf{D}\mathbf{x}^l \;\; \text{for} \;\; k \neq l \text{ and } ||\mathbf{x}^k|| = 1 \text{ for all } k \tag{1.27}$$

Problem 1.27 is called a *relaxation* of the original minimization problem 1.24. Intuitively, the solution of the relaxed problem is an approximation to the original problem 1.24 when the latter has a clustering with cost near the lower bound. This intuition has been proved formally by [4, 25]. Hence, spectral clustering algorithms are an approximate way to find the minimum *NCut*.

We have shown here that (1) when $\mathbf{P}$ is perfect, Algorithm 1.2 minimizes the *NCut* exactly and that (2) otherwise, the algorithm solves the relaxed problem 1.27 and *rounds* the results by K-means to obtain an approximately optimal *NCut* clustering.

1.3.3 Spectral clustering as finding smooth embeddings

Here we explore further the connection between the normalized cut of a clustering $\mathcal{C}$ and the Laplacian matrix $\mathbf{L}$ seen as an operator applied to functions on the set V, and the functional $||\mathbf{f}||^2_\Delta$ defined below as a **smoothness functional**.

Proposition 7. *Let* $\mathbf{L}$ *be normalized Laplacian defined by 1.7 and* $\mathbf{f} \in \mathbb{R}^n$ *be any vector indexed by the set of nodes* V*. Then*

$$\Delta f \;\stackrel{def}{=}\; \mathbf{f}^T \mathbf{L} \mathbf{f} \;=\; \sum_{ij \in E} S_{ij} \left(\frac{f_i}{\sqrt{d_i}} - \frac{f_j}{\sqrt{d_j}} \right)^2 \tag{1.28}$$

The proof follows closely the steps 1.19 to 1.20.

Now consider the *NCut* expression 1.24 and replace $\mathbf{y}^k$ by $\mathbf{D}^{-1/2}\mathbf{x}^k$ according to 1.21. We obtain[3]

$$\tilde{R}(\mathbf{y}^k) \;=\; R(\mathbf{x}^k) \;=\; \sum_{ij\in E} S_{ij}(x_i^k - \mathbf{x}_j^k)^2 \tag{1.29}$$

This shows that a clustering that has low *NCut* is one whose indicator functions $\mathbf{x}^{1:K}$ are *smooth w.r.t the graph* $\mathcal{G}$. In other words, the functions $\mathbf{x}^k$ must be almost constant on groups of nodes that are very similar, and are allowed to make abrupt changes only along edges $S_{ij} \approx 0$.

The symbol Δ and the name "Laplacian" indicate that $\mathbf{L}$ and $\mathbf{f}^T\mathbf{L}\mathbf{f}$ are the graph analogues of the well-known Laplace operator on $\mathbb{R}^d$, while Proposition 1.28 corresponds to the relationship $< f, \Delta f >= \int_{\mathrm{dom}\, f} |\nabla f|^2 dx$ in real analysis. The relationship between the continuous Δ and the graph Laplacian has been studied by [8, 14, 19].

1.4 Where do the similarities come from?

If the original data are vectors in $\mathbf{x}_i \in \mathbb{R}^d$ (note the abusive notation $\mathbf{x}$ in this section only), then the similarity is typically the *Gaussian kernel* (also called *heat kernel*)

$$S_{ij} \;=\; \exp\left(-\frac{||\mathbf{x}_i - \mathbf{x}_j||^2}{\sigma^2}\right) \tag{1.30}$$

This similarity gives raise to a complete graph $\mathcal{G}$, as $S_{ij} > 0$ always. Alternatively, one can define graphs that are dense only over *local neighborhoods*. For example, one can set S_{ij} by 1.30 if $||\mathbf{x}_i - \mathbf{x}_j|| \leq c\sigma$ and 0 otherwise, with the constant $c \approx 3$. This construction leads to a sparse graph, which is however a good approximation of the complete graph obtained by the heat kernel [44]. A variant of the above to zero out all S_{ij} except for the m nearest neighbors of data point i. This method used without checks can produce matrices that are not symmetric.

[3] This expression is almost identical to 1.20; the only difference is that in 1.20 the indicator vectors $\mathbf{x}^k$ take values in $\{0, 1\}$ while here they are normalized by $(\mathbf{x}^k)^T\mathbf{D}\mathbf{x}^k = 1$.

Even though the two graph construction methods appear to be very similar, it has been shown theoretically and empiricaly [19, 24] that the spectral clustering results they produce can be very different, both in high and in low dimensions. With the fixed m-nearest neighbor graphs, the clustering results are strongly favor balanced cuts, even if the cut occurs in regions of higher density; the radius-neighbor graph construction favors finding cuts of low density more. This is explained by the observation below, that the graph density in the latter graphs reflects the data density stronger than in the former type of graph.

It was pointed out that when the data density varies much, there is no unique radius that correctly reflects "locality", while the K-nearest neighbor graphs adapt to the variiying density. A simple and widely used way to "tune" the similarity function to the local density [49] is to set

$$S_{ij} \;=\; \exp\left(-\frac{||\mathbf{x}_i - \mathbf{x}_j||^2}{\sigma_i \sigma_j}\right) \tag{1.31}$$

where σ_i is the distance from $\mathbf{x}_i$ to its m-th nearest neighbor. Another simple heuristic to choose σ is to try various σ values and to pick the one that produces the smallest K-means cost in step 4 [34].

If the features in the data $\mathbf{x}$ have different units, or come from different modalities of measuring similarity, then it is useful to give each feature x_f a different kernel width σ_f. Hence, the similarity becomes

$$S_{ij} \;=\; \exp\left(\sum_{j=1}^{d} \frac{(x_{if} - x_{jf})^2}{\sigma_f^2}\right) \tag{1.32}$$

Clustering by similarities is not restricted to points in vector spaces. This represents one of the strengths of spectral clustering. If a distance $dist(i,j)$ can be defined on the data, then $dist(i,j)^2$ can substitute $||\mathbf{x}_i - \mathbf{x}_j||^2$ in 1.30; $dist$ can be obtained from the kernel trick [37]. Hence, spectral clustering can be applied to a variety of classes of non-vector data for which Mercer kernels have been designed, like trees, sequences or phylogenies [39, 13, 37].

Several methods for *learning* the similarities as a function of data features in a supervised setting exist [29],[4],[31]; the method of [31] has been extended to the unsupervised setting [40].

1.5 Practical considerations

The main advantage of spectral clustering is that it does not make any assumptions about the cluster shapes, and even allows clusters to "touch", as long as the clusters have sufficient overall separation and internal coherence (see e.g. Figure 1.2 right panels).

The method is computationally expensive compared to e.g center based clustering, as it needs to store and manipulate similarities/distances between all pairs of points instead of only distances to centers. The eigendecomposition step can also be computationally intensive. However, with a careful implementation, for example using sparse neighborhood graphs as in Section 1.4 instead of all pairwise similarities, and sparse matrix representations, the memory and computational requirements can be made tractable for sample sizes in the tens of thousands or larger. Several fast and approximate methods for spectral clustering have been proposed [11, 17, 23, 46].

It is known from matrix perturbation theory [42] that eigenvectors with smaller λ_k are more affected by numerical errors and noise in the similarities. This can be a problem when the number of clusters K is not small. In such a case, one can either (1) use only the first $K_0 < K$, eigenvectors of $\mathbf{P}$ or, (2) use the diffusion distance type rescaling $\mathbf{v}^k$ by λ_k^α, with $\alpha > 1$ which will smoothly decrease the effect of the noisier eigenvectors or (3) use Two-Way Spectral Clustering recursively.

One drawback of spectral clustering is the sensitivity of the eigenvectors $\mathbf{v}^k$ on the similarity $\mathbf{S}$ in ways that are not intuitive. For example, monotonic transformations of S_{ij}, even shift by a constant, can change a perfect $\mathbf{S}$ into one that is not perfect.

Outliers in spectral clustering need special treatment. An outlier is a point which has very low similarity with all other points (for example, because it is far away from them). An outlier will produce a spurious eigenvalue very close to 1 with an eigenvector which approximates an indicator vector for the outlier. So, l outliers in a data set will cause the l principal eigenvectors to be outliers, not clusters. Thus, it is *strongly recommended* that outliers be detected and removed *before* the eigendecomposition is performed. This is done easiest by removing all points for which $\sum_{j \neq i} S_{ij} \leq \epsilon$ for some ϵ which is small w.r.t. the average d_i. Also before the eigendecomposition, one should detect if $\mathcal{G}$ is disconnected by a *connected components* algorithm (see Chapter **??**).

1.6 Conclusions and further reading

The tight relationship between K-MEANS and SPECTRALCLUSTERING hints at the situations when SPECTRALCLUSTERING is recommended. Namely, SPECTRALCLUSTERING returns hard, non-overlaping clusterings, requires the number of clusters K as input, and works best when this number is not too large (up to $K = 10$). For larger K, recursive partitioning based on TWO-WAY SPECTRAL CLUSTERING is more robust.The relationship with K-MEANS is even deeper than we have presented it here [16]. As mentioned above, the algorithm is sensitive to outliers and transformations of $\mathbf{S}$, but it is very robust to the shapes of clusters, to small amounts of data "spilling" from one cluster to the another, and can balance well cluster sizes and their internal coherence.

For chosing the number of clusters K, there are two important indicators: the *eigengap* $\lambda_K - \lambda_{K+1}$, and the *gap* $NCut(\mathcal{C}_K) - (K - \sum_{k=1}^{K} \lambda_K)$, where we have denoted by $\mathcal{C}_K$ the clustering returned by a spectral clustering algorithm with input $\mathbf{S}$ and K. Ideally, the former should be large, indicating a stable principal subspace, and the latter should be near zero, indicating almost perfect $\mathbf{P}$ for that K and $\mathcal{C}_K$. A heuristic proposed by [26] is to find the knee in the graph of gap vs. K, or in the graph of gap divided by the eigengap, as suggested by the theory in [25]; [3] proposes heuristic based on the eigengaps $\lambda_k^t - \lambda_{k+1}^t$ for $t > 1$ that can find clusterings at different granularity levels and works well for matrices that are almost block diagonal.

Other formulations of clustering that aim to minimize the same Normalized Cut criterion are based on Semidefinite Programming [48], and on submodular function optimization [33, 9, 22].

Spectral clustering has been extended to directed graphs [36, 2, 28] as well as finding *the local cluster* of a data point in a large graph [41]

Clusterability for spectral clustering, i.e. the problem of defining what is a “good” clustering, has been studied by [27, 25, 1, 6, 21]; some of these references also introduced new algorithms with guarantees that depend on how clusterable is the data.

Finally, the ideas and algorithms presented here have deep connections with the fast growing areas of *non-linear dimension reduction*, also known as *manifold learning* [8] and of solving very large linear systems [7].

References

[1] Margareta Ackerman and Shai Ben-David. Clusterability: A theoretical study. In David A. Van Dyk and Max Welling, editors, *Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics, AISTATS 2009, Clearwater Beach, Florida, USA, April 16-18, 2009*, volume 5 of *JMLR Proceedings*, pages 1–8. JMLR.org, 2009.

[2] Reid Andersen, Fan R. K. Chung, and Kevin J. Lang. Local partitioning for directed graphs using pagerank. In *WAW*, pages 166–178, 2007.

[3] Arik Azran and Zoubin Ghahramani. Spectral methods for automatic multiscale data clustering. In *Computer Vision and Pattern Recognition*, pages 190–197. IEEE Computer Society, 2006.

[4] Francis Bach and Michael I. Jordan. Learning spectral clustering with applications to speech separation. *Journal of Machine Learning Research*, 7:1963–2001, 2006.

[5] Sivaraman Balakrishnan, Min Xu, Akshay Krishnamurthy, and Aarti Singh. Noise thresholds for spectral clustering. In *Advances in Neural Information Processing Systems 24: 25th Annual Conference on Neural Information Processing Systems 2011. Proceedings of a meeting held 12-14 December 2011, Granada, Spain.*, pages 954–962, 2011.

[6] Maria-Florina Balcan and Mark Braverman. Finding low error clusterings. In *COLT 2009 - The 22nd Conference on Learning Theory, Montreal, Quebec, Canada, June 18-21, 2009*, 2009.

[7] Joshua D. Batson, Daniel A. Spielman, Nikhil Srivastava, and Shang-Hua Teng. Spectral sparsification of graphs: theory and algorithms. *Commun. ACM*, 56(8):87–94, 2013.

[8] M. Belkin and P. Niyogi. Laplacian eigenmaps and spectral techniques for embedding and clustering. In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, Cambridge, MA, 2002. MIT Press.

[9] Yuri Boykov, Olga Veksler, and Ramin Zabih. Fast approximate energy minimization via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(11):1222–1239, 2001.

[10] Sebastien Bubeck, Marina Meilă, and Ulrike von Luxburg. How the initialization affects the stability of the k-means algorithm. *ESAIM: Probability and Statistics*, 16:436–452, 2012.

[11] Bo Chen, Bin Gao, Tie-Yan Liu, Yu-Fu Chen, and Wei-Ying Ma. Fast spectral clustering of data using sequential matrix compression. In *Proceedings of the 17th European Conference on Machine Learning, ECML*, pages 590–597, 2006.

[12] Fan R. K. Chung. *Spectral Graph Theory.* Number Regional Conference Series in Mathematics in 92. American Mathematical Society, Providence, RI, 1997.

[13] Alexander Clark, Christophe Costa Florêncio, and Chris Watkins. Languages as hyperplanes: grammatical inference with string kernels. *Machine Learning*, 82(3):351–373, 2011.

[14] R. R. Coifman and S. Lafon. Diffusion maps. *Applied and Computational Harmonic Analysis*, 30(1):5–30, 2006.

[15] Sanjoy Dasgupta and Leonard Schulman. A probabilistic analysis of em for mixtures of separated, spherical gaussians. *Journal of Machine Learnig Research*, 8:203–226, Feb 2007.

[16] Chris Ding and Xiaofeng He. K-means clustering via principal component analysis. In Carla E. Brodley, editor, *Proceedings of the International Machine Learning Conference (ICML)*. Morgan Kauffman, 2004.

[17] Charles Fowlkes, Serge Belongie, Fan Chung, and Jitendra Malik. Spectral grouping using the Nyström method. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(2):214–225, 2004.

[18] Yoram Gdalyahu, Daphna Weinshall, and Michael Werman. Stochastic image segmentation by typical cuts. In *Computer Vision and Pattern Recognition*, volume 2, page 2596. IEEE Computer Society, IEEE, 1999.

[19] Matthias Hein, Jean-Yves Audibert, and Ulrike von Luxburg. Graph laplacians and their convergence on random neighborhood graphs. *Journal of Machine Learning Research*, 8:1325–1368, 2007.

[20] Dorit S. Hochbaum and David B. Shmoys. A best possible heuristic for the k-center problem. *Mathematics of Operations Research*, 10(2):180–184, May 1985.

[21] Ravi Kannan, Santosh Vempala, and Adrian Vetta. On clusterings: good, bad and spectral. In *Proc. of 41st Symposium on the Foundations of Computer Science, FOCS 2000*, 2000.

[22] V. Kolmogorov and R. Zabih. What energy functions can be minimized via graph cuts? *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(2):147–159, 2004.

[23] Tie-Yan Liu, Huai-Yuan Yang, Xin Zheng, Tao Qin, and Wei-Ying Ma. Fast large-scale spectral clustering by sequential shrinkage optimization. In *Proceedings of the 29th European Conference on IR Research*, pages 319–330, 2007.

[24] Markus Maier, Ulrike von Luxburg, and Matthias Hein. Influence of graph construction on graph-based clustering measures. In *Advances in Neural Information Processing Systems 21, Proceedings of the Twenty-Second Annual Conference on Neural Information Processing Systems, Vancouver, British Columbia, Canada, December 8-11, 2008*, pages 1025–1032, 2008.

[25] Marina Meila. The stability of a good clustering. Technical Report 624, University of Washington, June 2014.

[26] Marina Meila and Liang Xu. Multiway cuts and spectral clustering. Technical Report 442, University of Washington, Department of Statistics, May 2003.

[27] Marina Meilă. The uniqueness of a good optimum for K-means. In Andrew Moore and William Cohen, editors, *Proceedings of the International Machine Learning Conference (ICML)*, pages 625–632. International Machine Learning Society, 2006.

[28] Marina Meilă and William Pentney. Clustering by weighted cuts in directed graphs. In *Proceedings of the Seventh SIAM International Conference on Data Mining, April 26-28, 2007, Minneapolis, Minnesota, USA*, pages 135–144, 2007.

[29] Marina Meilă and Jianbo Shi. Learning segmentation by random walks. In T. K. Leen, T. G. Dietterich, and V. Tresp, editors, *Advances in Neural Information Processing Systems*, volume 13, pages 873–879, Cambridge, MA, 2001. MIT Press.

[30] Marina Meilă and Jianbo Shi. A random walks view of spectral segmentation. In T. Jaakkola and T. Richardson, editors, *Artificial Intelligence and Statistics AISTATS*, 2001.

[31] Marina Meilă, Susan Shortreed, and Liang Xu. Regularized spectral learning. In Robert Cowell and Zoubin Ghahramani, editors, *Proceedings of the Artificial Intelligence and Statistics Workshop(AISTATS 05)*, 2005.

[32] Boaz Nadler, Stephane Lafon, Ronald Coifman, and Ioannis Kevrekidis. Diffusion maps, spectral clustering and eigenfunctions of fokker-planck operators. In Y. Weiss, B. Schölkopf, and J. Platt, editors, *Advances in Neural Information Processing Systems 18*, pages 955–962, Cambridge, MA, 2006. MIT Press.

[33] Mukund Narasimhan and Jeff Bilmes. Local search for balanced submodular clusterings. In Manuela M. Veloso, editor, *IJCAI 2007, Proceedings of the 20th International Joint Conference on Artificial Intelligence, Hyderabad, India, January 6-12, 2007*, pages 981–986, 2007.

[34] A. Y. Ng, M. I. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, Cambridge, MA, 2002. MIT Press.

[35] Christos Papadimitriou and Kenneth Steiglitz. *Combinatorial optimization. Algorithms*

and complexity. Dover Publication, Inc., Minneola, NY, 1998.

[36] William Pentney and Marina Meilă. Spectral clustering of biological sequence data. In Manuela Veloso and Subbarao Kambhampati, editors, *Proceedings of Twentieth National Conference on Artificial Intelligence (AAAI-05)*, pages 845–850, Menlo Park, California, 2005. The AAAI Press.

[37] Bernhard Schölkopf and Alexander J. Smola. *Learning with Kernels.* M. I. T. Press, Cambridge, MA, 2002.

[38] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *PAMI*, 2000.

[39] Kilho Shin, Marco Cuturi, and Tetsuji Kuboyama. Mapping kernels for trees. In Lise Getoor and Tobias Scheffer, editors, *Proceedings of the 28th International Conference on Machine Learning, ICML 2011, Bellevue, Washington, USA, June 28 - July 2, 2011*, pages 961–968. Omnipress, 2011.

[40] Susan Shortreed and Marina Meilă. Unsupervised spectral learning. In Tommi Jaakkola and Fahiem Bachhus, editors, *Proceedings of the 21st Conference on Uncertainty in AI*, pages 534–544, Arlington,Virginia, 2005. AUAI Press.

[41] Daniel Spielman and Shang-Hua Teng. A local clustering algorithm for massive graphs and its application to nearly-linear time graph partitioning. Technical Report :0809.3232v1 [cs.DS], arXiv, 2008.

[42] Gilbert W. Stewart and Ji-guang Sun. *Matrix perturbation theory.* Academic Press, San Diego, CA, 1990.

[43] G. Strang. *Linear Algebra and its applications, 3rd Edition.* Saunders College Publishing, 1988.

[44] Daniel Ting, Ling Huang, and Michael I. Jordan. An analysis of the convergence of graph laplacians. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10), June 21-24, 2010, Haifa, Israel*, pages 1079–1086, 2010.

[45] Deepak Verma and Marina Meilă. A comparison of spectral clustering algorithms. TR 03-05-01, University of Washington, May 2003.

[46] Fabian Wauthier, Nebojsa Jojic, and Michael Jordan. Active spectral clustering via iterative uncertainty reduction. In *18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1339–1347, 2012.

[47] Scott White and Padhraic Smyth. A spectral clustering approach to finding communities in graphs. In *Proceedings of SIAM International Conference on Data Mining*, 2005.

[48] Eric P. Xing and Michael I. Jordan. On semidefinite relaxation for normalized k-cut and connections to spectral clustering. Technical Report UCB/CSD-03-1265, EECS Department, University of California, Berkeley, Jun 2003.

[49] Lihi Zelnik-Manor and Pietro Perona. Self-tuning spectral clustering. In *Advances in Neural Information Processing Systems*, volume 17, pages 1601–1608, 2004.