Information Processing and Management 51 (2015) 616–624

Contents lists available at ScienceDirect

Information Processing and Management

journal homepage: www.elsevier.com/locate/infoproman

Large-scale spectral clustering based on pairwise constraints

T. Semertzidis [a,b], D. Rafailidis [a,*], M.G. Strintzis [a,b], P. Daras [a]

[a] Information Technologies Institute, Centre for Research and Technology Hellas, Thessaloniki, 6th km Charilaou – Thermi, 57001, P.O. Box 60361, Greece
[b] Electrical and Computer Engineering Department, Aristotle University of Thessaloniki, Thessaloniki, Greece

ARTICLE INFO

Article history:
Received 26 August 2014
Received in revised form 24 January 2015
Accepted 20 May 2015
Available online 12 June 2015

Keywords:
Spectral clustering
Semi-supervised
Sparse coding

ABSTRACT

In this paper, we present an efficient spectral clustering method for large-scale data sets, given a set of pairwise constraints. Our contribution is threefold: (a) clustering accuracy is increased by injecting prior knowledge of the data points' constraints to a small affinity submatrix; (b) connected components are identified automatically based on the data points' pairwise constraints, generating thus isolated "islands" of points; furthermore, local neighborhoods of points of the same connected component are adapted dynamically, and constraints propagation is performed so as to further increase the clustering accuracy; finally (c) the complexity is preserved low, by following a sparse coding strategy of a landmark spectral clustering. In our experiments with three benchmark shape, face and handwritten digit image data sets, we show that the proposed method outperforms competitive spectral clustering methods that either follow semi-supervised or scalable strategies.

© 2015 Elsevier Ltd. All rights reserved.

1. Introduction

Spectral Clustering (SC) is a popular approach for solving clustering problems in a wide range of non-Euclidean spaces, linearly non-separable clusters and detecting non-convex patterns (Filippone, Camastra, Masulli, & Rovetta, 2008). SC methods are used in numerous real-world applications such as image segmentation (Tung, Wong, & Clausi, 2010), face recognition (Cevikalp & Triggs, 2010), feature fusion (Huang, Chuang, & Chen, 2012), speech recognition (Iso, 2010), 3D shape retrieval (Tatsuma & Aono, 2009) and protein sequences clustering (Paccanaro, Chennubhotla, Casbon, & Saqi, 2003). The key idea in SC is to achieve graph partitioning by performing eigendecomposition of a graph Laplacian matrix. The SC approach is formulated as follows: given a set of d-dimensional data points[1] $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\} \in \mathbb{R}^d$, SC methods construct an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, represented by the $W \in \mathbb{R}^{n \times n}$ affinity matrix (or the respective adjacency), where $\mathcal{V}$ and $\mathcal{E}$ are the sets of vertices and edges, respectively. The goal is to find a k-way partitioning, i.e. k disjoint data subsets whose union is the whole data set, to minimize a particular objective. SC methods firstly calculate the degree matrix $D = \sum_j W_{ji} \in \mathbb{R}^{n \times n}$, a diagonal matrix whose entries are column (or row, since W is symmetric) sums of W. Then, a Laplacian matrix is constructed and its eigenvectors are used as the low k-th dimensional representation of the data. Finally, the k-means algorithm is applied to generate the clusters.

Spectral clustering methods differ in how they define and construct the Laplacian matrix and thus which eigenvectors are selected to represent the partitioning, aiming to exploit special properties of different matrix formulations (Filippone et al.,

[*] Corresponding author.
E-mail addresses: theosem@iti.gr (T. Semertzidis), drafail@iti.gr (D. Rafailidis), strintzi@eng.auth.gr (M.G. Strintzis), daras@iti.gr (P. Daras).

[1] Following standard notations, we use capital italic letters for matrices (e.g. A), lower-case bold letters for vectors (e.g. $\mathbf{a}$) and calligraphic fonts for sets (e.g. $\mathcal{A}$).

http://dx.doi.org/10.1016/j.ipm.2015.05.007
0306-4573/© 2015 Elsevier Ltd. All rights reserved.

2008; Luxburg, 2007). For the interested reader, Ulrike von Luxburg's tutorial (Luxburg, 2007) includes examples of different Laplacians' constructions. Moreover, different objective functions are used to derive the best cut. For example, Ratio Cut (Chan, Schlag, & Zien, 1993) tries to minimize the total cost of the edges crossing the cluster boundaries, normalized by the size of the k clusters, to encourage balanced cluster sizes. Normalized Cut (NCut) (Shi & Malik, 1997) uses the same objective criterion as Ratio Cut, normalized by the total degree of each cluster, making thus the clusters having similar degrees.

However, irrespective of the selected approach, there are two important factors for applying a SC method to a real world application: (a) the scalability of the method to large datasets; and (b) the high clustering accuracy.

1.1. Problem definition and current solutions

Baseline SC methods (Chan et al., 1993; Shi & Malik, 1997) require (a) $O(n^2)$ time to calculate the W affinity matrix and consequently to construct the graph $\mathcal{G}$ and the Laplacian matrix L; and (b) $O(n^3)$ time to calculate the eigendecomposition of L. Both complexities prohibit the direct application of SC for generating clusters in large-scale data sets. Several accelerated methods (Cao, Chen, Dai, & Ling, 2014; Chen & Cai, 2011; Fowlkes, Belongie, Chung, & Malik, 2004; Liu, Wang, Danilevsky, & Han, 2013; Yan, Huang, & Jordan, 2009) have been proposed in the literature trying to reduce the initial problem size of n data points by selecting p ($\ll n$) samples of the data set.[2] Accelerated methods in their approximations calculate $n \times p$ distances to construct $\mathcal{G}$ and perform the eigendecomposition to a highly reduced $L \in \mathbb{R}^{p \times p}$ Laplacian matrix. Consequently, accelerated methods significantly decrease the high complexity of the baseline SC methods (Chung, 1997; Ng, Jordan, & Weiss, 2002).

Nevertheless, with respect to the clustering accuracy, accelerated methods either fail in their approximations for a low number of p samples, or do not overcome the limited accuracy of the baseline SC methods. Baseline SC methods tend to unbalanced clusters, i.e. single nodes are separated from the rest of the graph. As a result they are noise-sensitive, i.e. few isolated points can easily draw the cuts away from the global partitions (Chang & Yeung, 2008). Additionally, baseline SC methods cannot exploit the information of a set of data points' pairwise constraints, in order to increase the clustering accuracy, since they function in an unsupervised manner. Several works have extended SC in a semi-supervised way (Chen & Feng, 2012; Kulis, Basu, Dhillon, & Mooney, 2009; Wagstaff, Cardie, Rogers, & Schroedl, 2001), where the goal is to incorporate prior information into the algorithm, in order to improve the clustering results. This is achieved by adding a preprocessing step, where pairwise must-link (pairs of points that should belong to the same cluster) and cannot-link constraints (pairs of points that should belong to different clusters) are added to the $W \in \mathbb{R}^{n \times n}$ affinity matrix. Semi-supervised SC methods achieve higher clustering accuracy compared to conventional SC methods. However, existing semi-supervised SC methods preserve the high complexity of the baseline SC methods and thus are not scalable.

1.2. Contribution and layout

The contribution of the proposed method is threefold: (a) the clustering accuracy is increased by injecting prior knowledge of the data points' constraints to a small affinity submatrix; (b) according to the Tarjan's algorithm (Tarjan, 1972) connected components (CC) are automatically identified from the data points' constraints, generating thus isolated "islands" of points. Then, for each CC the local neighborhood of points is adapted dynamically and constraints propagation is performed so as to increase the clustering accuracy; finally (c) the complexity is preserved low, by following a landmark spectral clustering strategy to ensure scalability. In our experiments with three benchmark face, shape and handwritten image data sets, we show that the proposed method outperforms state-of-the-art spectral clustering methods that either follow semi-supervised or scalable strategies in terms of clustering accuracy and computational cost.

The rest of the paper is organized as follows: in Section 2 the proposed method is described in detail. In Section 3 the experimental results are presented and discussed, finally, in Section 4 the conclusions of this study are drawn.

2. Proposed method

Given a set of p points that participate in the pairwise constraints, the proposed method consists of the following steps: (a) generate a sparse representation of an affinity submatrix $\widehat{Z} \in \mathbb{R}^{p \times n}$, expressing the similarities between the p points that participate in the pairwise constraints and the whole data set n; (b) compute a temporal $p \times p$similarity/adjacency matrix based only on the p points that participate in the pairwise constraints to extract the connected components automatically, generating thus isolated "islands" of the p points; (c) update the sparse affinity submatrix $\widehat{Z}$ based on the p points co-appearances to the same or different connected components; (d) perform dynamical adaptation of the p data points' local neighborhood in each connected component; (e) propagate constraints to the local neighboring points in the connected component; and (f) perform eigedecomposition of a highly reduced matrix and apply k-means to generate the final k clusters.

[2] Additionally, several methods perform parallel SC in distributed systems (Chen, Song, Bai, Lin, & Chang, 2011; Kang, Meeder, Papalexakis, & Faloutsos, 2014) to reduce the computational time of SC.

Formally, given a set of d-dimensional data points $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\} \in \mathbb{R}^d$, denoted by a $X \in \mathbb{R}^{d\times n}$ matrix; and a subset $\mathcal{T}$ of p points, with $(p \ll n)$ denoting the points that participate in the sets of $\mathcal{M}$ must-link or $\mathcal{C}$ cannot-link constraints (with the constraints' pool being equal to $\mathcal{M} \cup \mathcal{C}$), the goal is to partition the n points into k discrete clusters, with the boundaries of the k clusters lying afar. The final goal is to design the $W \in \mathbb{R}^{n\times n}$ affinity matrix as $W = \widehat{Z}^T\widehat{Z}$, where $\widehat{Z} \in \mathbb{R}^{p\times n}$ is the p-th dimensional representation of the n data points, expressed as similarities/affinities of the n data points to the p points that participate in the pairwise constraints. The $X \in R^{d\times n}$ matrix can be approximated as $X \approx UZ$, where the columns of matrix $U \in \mathbb{R}^{d\times p}$ are called basis vectors, i.e. the d-dimensional vectors of the p points. Therefore, the goal is to minimize the approximation error $\min_{U,Z}\|X - UZ\|^2$, where $\|\cdot\|$ denotes the Frobenius norm of a matrix.

2.1. Sparse representation of the affinity submatrix

Following the sparse coding strategy of (Chen & Cai, 2011), based on the Nadaraya-Watson kernel regression (Härdle, 1992), for any data point $\mathbf{x}_i$ its $\widehat{\mathbf{x}}_i$ approximation is calculated as:

$$\widehat{\mathbf{x}}_i = \sum_{j=1}^{p} z_{ji}\mathbf{u}_j \tag{1}$$

where $\mathbf{u}_j$ is the j-th column vector of U and z_{ji} is the ji-th element of Z. Then, to create the sparse representation of the Z affinity sparse matrix, the z_{ji} value is set to 0, if $\mathbf{u}_j$ is not among the $r \leqslant p$ nearest points.[3] Let $\langle I\rangle \in \mathbb{R}^{d\times r}$ denote a submatrix of U, composed of r nearest constrained points of $\mathbf{x}_i$. Then, each element z_{ji} is computed as:

$$z_{ji} = \frac{\Phi(\mathbf{x}_i, \mathbf{u}_j)}{\sum_{j'\in\langle I\rangle}\Phi(\mathbf{x}_i, \mathbf{u}_{j'})}, \qquad i \in 1\ldots n \text{ and } j \in \langle I\rangle \tag{2}$$

where $\Phi(\cdot)$ is a kernel function with bandwidth σ. The Gaussian kernel $\Phi(\mathbf{x}_i, \mathbf{u}_j) = \exp(-\|\mathbf{x}_i - \mathbf{u}_j\|/2\sigma^2)$ is one of the most commonly used, where σ controls the local scale of each data point's neighborhood. Therefore, based on (2), the $Z \in \mathbb{R}^{p\times n}$ sparse representation is calculated. Consequently, for the W affinity matrix it holds that $W = \widehat{Z}^T\widehat{Z}$, where $\widehat{Z} = D^{-1/2}Z$ is the normalized Z by the $D = \sum_j Z_{ji}$ degree matrix.

2.2. Connected components' extraction from pairwise constraints

Next, we retrieve all the p distinct points that participate in the $\mathcal{M}$ must-link and the $\mathcal{C}$ cannot-link sets of constraints. We generate a $p \times p$ adjacency matrix, with 1s and 0s for the must-link $\mathcal{M}$ and cannot-link $\mathcal{C}$ constraints, respectively. According the calculated $p \times p$ adjacency matrix the Tarjan's algorithm[4] is used to detect the $\mathcal{H}$ strongly connected components of the p points that participate in the pairwise constraints, generating thus $\mathcal{H}$ isolated "islands" of the p points. The identified connected components are used to set the initial constraints to the $\widehat{Z}$ affinity submatrix.

Let $\mathcal{T} \equiv \{\mathcal{T}_1 \cup \mathcal{T}_2, \ldots, \cup\mathcal{T}_a\cup, \ldots, \cup\mathcal{T}_{|\mathcal{H}|}\}$, where $\mathcal{T}_a$ is the set of points of the a-th connected component, with $a \in \{1, \ldots, |\mathcal{H}|\}$ and $|\mathcal{T}| = p$. Let $\mathcal{T}'_a$ be the relative complement of $\mathcal{T}_a$ in $\mathcal{T}$ with $\mathcal{T}'_a \equiv \mathcal{T} \setminus \mathcal{T}_a \equiv \{p'_a \in \mathcal{T} | p'_a \notin \mathcal{T}_a\}$. Since the similarities between the $p_a \in \mathcal{T}_a$ points of the a-th connected component should be maximum and the similarities between the $p'_a \in \mathcal{T}'_a$ and $p_a \in \mathcal{T}_a$ points of different connected components should be minimum, we set the $\binom{|\mathcal{T}|}{2}$ constraints to the $\widehat{Z}$ affinity submatrix as follows:

$$\widehat{Z}(p_i, p_j) = \begin{cases} 1, & \{p_i \in \mathcal{T}_a | p_j \in \mathcal{T}_a\} \\ 0, & \{p_i \in \mathcal{T}_a | p_j \in \mathcal{T}'_a\} \end{cases} \tag{3}$$

In other words, we set the affinity matrix elements to 1 when points belong to the same connected component or 0 if points are in different connected components.

2.3. Constraints propagation to points' local neighborhoods

Then, we generate $|\mathcal{H}|$ different adjacency submatrices $L_{\mathcal{T}_a} \in \mathbb{R}^{l\times|\mathcal{T}_a|}$, where for each distinct connected component a each submatrix $L_{\mathcal{T}_a}$ contains a subset $|\mathcal{T}_a| < p$ of points that participate in the pairwise constraints and the l neighbors of the $|\mathcal{T}_a|$ points based on the affinity submatrix $\widehat{Z}$. Then, $\forall L_{\mathcal{T}_a}$ we calculate the appearances $(freq_i)$ of each neighbor i, with $i \in \{1, \ldots, l\}$. The calculated $freq_i$ frequencies are sorted in ascending order to create a m-th dimensional vector $\mathbf{degreeVec} = (freq_{1'}, freq_{2'}, \ldots, freq_{m'})$, with $freq_{1'} \leqslant freq_{2'} \leqslant \cdots freq_{m'}$, where m is the number of unique frequencies of the neighbors of $L_{\mathcal{T}_a}$ and $freq_{i'}$ is a sorted $freq_i$ frequency. The neighbors in $L_{\mathcal{T}_a}$ with high appearances in the local neighborhood

[3] This holds because z_{ji} should be larger if $\mathbf{x}_i$ is closer to $\mathbf{u}_j$.

[4] The Tarjan's algorithm (Tarjan, 1972) generates connected components automatically from an affinity/adjacency matrix.

should be the closest neighbors of the $p_a \in \mathcal{T}_a$ points of the a-th component. Following a linear interpolation strategy, we generate a m-th dimensional vector **lookupVec** as follows:

$$\mathbf{lookupVec}_{i'} = \begin{cases} \min + freq_{i'} \times \frac{\max - \min}{m-1}, & m \neq 1 \\ \max, & m = 1 \end{cases} \tag{4}$$

where min and max are the minimum and maximum similarities in $L_{\mathcal{T}_a}$, respectively and $i' \in \{1, \ldots, m\}$. Neighbors of low appearances should lie far (min) from the $p_a \in \mathcal{T}_a$ points of component a and close (max) in case of high appearances.

Let $\mathcal{L}_{pi}$ and $\mathcal{L}_{pj}$ be respectively the set of the l nearest neighbors of p_i and p_j, with $p_i, p_j \in \mathcal{T}_a$ and $i, j \in \{1, \ldots, |\mathcal{T}_a|\}$. Due to the extreme sparsity in the $\widehat{Z}$ affinity submatrix, the following problem arises: in many cases it holds $\mathcal{L}_{pi} \cap \mathcal{L}_{pj} \equiv \emptyset$, i.e. points p_i and p_j of the same a-th connected component do not have common neighbors. Therefore, $\forall\, p_i \in \mathcal{T}_a$ we propagate the neighbors of the rest $p_j \in \mathcal{T}_a$ points as follows:

$$\mathcal{L}_{pi} = \{\mathcal{L}_{p1} \cup \mathcal{L}_{p2} \cup \ldots \cup \mathcal{L}_{p|\mathcal{T}a|}\}, \text{ with } i \in 1, \ldots, |\mathcal{T}_a| \tag{5}$$

Then, based on (4) and (5) we propagate the constraints to the $\widehat{Z}$ affinity submatrix as follows:

$$\widehat{Z}(i, p_j) = \mathbf{lookupVec}_{i'} \tag{6}$$

where $i \in \mathcal{L}_{pj}, p_j \in \mathcal{T}_a$ and $freq_i = freq_{i'}$, with i being neighbor of p_j; point p_j belonging to the a-th connected component according to the $\mathcal{M} \cup \mathcal{C}$ constraints pool; and frequency ($freq_i$) of neighbor i (lying into the local neighborhood of points $p_j \in \mathcal{T}_a$) being equal to the sorted frequency $freq_{i'}$, stored in vector **degreeVec**.

Next, we present an example of the constraints propagation method. Let $\mathcal{T}_a = \{p_1, p_2\}$ be a connected component, with $\widehat{Z}(p_1, p_2) = 1$ and $|\mathcal{T}_a| = 2$. Also, let $\mathcal{L}_{p_1} = \{x_1, x_2\}$ and $\mathcal{L}_{p_2} = \{x_1, x_3\}$ be the sets of neighbors of p_1 and p_2, with $l = 2$. Given $\widehat{Z}(x_1, p_1) = 0.4$, $\widehat{Z}(x_2, p_1) = 0.6$ and $\widehat{Z}(x_1, p_2) = 0.5$, $\widehat{Z}(x_3, p_2) = 0.7$, we have 3 unique neighbors in the connected component $\mathcal{T}_a$, with x_1 appearing 2 times and x_2, x_3 1 time. In this example, we have $m = 2$ unique frequencies, generating thus vector $\mathbf{degreeVec} = (1, 2)$. Then, based on **degreeVec** and (4) with min = 0.4 and max = 0.7, i.e. the minimum and the maximum similarities of the neighbors of p_1 and p_2 in the connected component $\mathcal{T}_a$, we generate the **lookupVec** vector as follows:

$$\mathbf{lookupVec} = \left(0.4 + \mathbf{degreeVec}_1 \times \frac{(0.7 - 0.4)}{2 - 1}, 0.4 + \mathbf{degreeVec}_2 \times \frac{(0.7 - 0.4)}{2 - 1}\right) \Longleftrightarrow \mathbf{lookupVec} = (0.7, 1)$$

According to (5), we propagate the neighbors of p_1 and p_2 as follows: $\mathcal{L}_{p_1} \equiv \mathcal{L}_{p_2} \equiv \{x_1, x_2, x_3\}$. Finally, based on the new sets of neighbors $\mathcal{L}_{p_1}, \mathcal{L}_{p_2}$ and Eq. (6), the $\widehat{Z}$ affinity submatrix is updated: $\widehat{Z}(x_1, p_1) = 1$, $\widehat{Z}(x_2, p_1) = 0.7$, $\widehat{Z}(x_3, p_1) = 0.7$ and $\widehat{Z}(x_1, p_2) = 1$, $\widehat{Z}(x_2, p_2) = 0.7$, $\widehat{Z}(x_3, p_2) = 0.7$.

2.4. Clusters' generation

Let the Singular Value Decomposition (SVD) of $\widehat{Z} = A\Sigma B^T$, where $\Sigma = diag(\sigma_1, \ldots, \sigma_p)$ and $\sigma_1 \geqslant \sigma_2 \geqslant \ldots \geqslant \sigma_p \geqslant 0$ are the singular values of $\widehat{Z}, A = [\mathbf{a}_1, \ldots, \mathbf{a}_p] \in \mathbb{R}^{p \times p}$ and $\mathbf{a}_i$'s are called left singular vectors, $B = [\mathbf{b}_1, \ldots, \mathbf{b}_p] \in \mathbb{R}^{n \times p}$ and $\mathbf{b}_i$'s are called right singular eigenvectors. It is easy to verify that B are the eigenvectors of matrix $\widehat{Z}^T\widehat{Z}$ and A are the eigenvectors of matrix $\widehat{Z}\widehat{Z}^T$. Since the size of $\widehat{Z}\widehat{Z}^T$ is $p \times p$, we can compute A in $O(p^3)$ and then B can be computed as $B = \Sigma^{-1}A^T\widehat{Z}$. The overall time is $O(p^3 + p^2 n)$, significantly reduced than $O(n^3)$ since $p \ll n$. In order to obtain the final k clusters the traditional k-means method is applied to the n right singular eigenvectors $\mathbf{b}_i$'s, i.e. the rows of B.

2.5. Complexity analysis

The algorithmic steps of the proposed Spectral Clustering method based on Pairwise Constraints (SC-PC) are described in Algorithm 1 with reference to the corresponding equations.

Given n data points with dimensionality d, we use $p \ll n$ pairwise-constrained points that form $|\mathcal{H}|$ different connected components based on the constraints pool $\mathcal{M} \cup \mathcal{C}$. The total complexity of the proposed method is the sum of: (a) $O(|\mathcal{M}| + |\mathcal{C}|)$ to compute the connected components of the p points participating in the $|\mathcal{M}|$ must-link and $|\mathcal{C}|$ cannot link constraints; (b) $O(pnd)$ to compute the $\widehat{Z}$ affinity submatrix (graph construction); (c) $O(l|\mathcal{H}|)$ to create the $|\mathcal{H}|$ adjacency submatrices and to perform the constraints propagation to the local neighborhoods of each component; (d) $O(p^3 + p^2 n)$ to compute the eigenvectors of B; and (e) $O(tknp)$ to perform the traditional k-means to the n right singular p-dimensional eigenvectors $\mathbf{b}_i$'s, where t is the number of iterations in k-means.

Algorithm 1. SC-PC algorithm

```
Require: dataset $X \in R^{d\times n}$, set of pairwise constraints $\mathcal{M} \cup \mathcal{C}$, points $p \in \mathcal{T}$ that participate in the pairwise constraints ($p \ll n$)
Ensure: $k$ clusters
  Compute $\widehat{Z} \in \mathbb{R}^{p\times n}$ (Eqs. (2) and (3))
  Compute Connected Components (CC) of points $p \in \mathcal{T} \subset X$
  for each CC do
    Generate the respective adjacency submatrix $L_{\mathcal{T}_a}$ for the $a$-th CC
    Build lookup table with neighbor frequencies (Eq. (4))
    Constraints propagation to points of the same CC (Eq. (5))
    Update of $\widehat{Z}$ matrix with lookup table values (Eq. (6))
  end for
  Compute the first $k$ eigenvectors of $Z\widehat{Z}$, denoted by $A = [\mathbf{a}_1, \ldots, \mathbf{a}_p] \in \mathbb{R}^{p\times p}$
  Compute $B = [\mathbf{b}_1, \ldots, \mathbf{b}_p] \in \mathbb{R}^{n\times p}$
  Each row of $B$ is a data point and apply k-means to get the clusters
```

3. Experimental results

3.1. Data sets

In our experiments we used three high-dimensional benchmark data sets,[5] including a shape image data set (COIL100, Nene, Nayar, & Murase, 1996), a face data set (CMU PIE, Bsat, Baker, & Sim, 2001) and a handwritten digit data set (MNIST, Cun, Bottou, Bengio, & Haffner, 1998). Table 1 summarizes the details of the data sets.

3.2. Compared algorithms

The proposed Spectral Clustering method based on Pairwise Constraints (SC-PC), was compared against (a) the baseline NCut method (Ng et al., 2002); (b) the semi-supervised method of Near Strangers or Distant Relatives (NSDR) (Chen & Feng, 2012); the accelerated methods of (c) Nyström approximation-based SC (Fowlkes et al., 2004); (d) LSC-K (Chen & Cai, 2011); (e) LSC-R (Chen & Cai, 2011); and (f) LSC-WPR (Rafailidis et al., 2014).

In (Chen & Feng, 2012), Chen and Feng proposed the semi-supervised method of NSDR. Provided a set of per class must-link and cannot-link constraints to a $n \times n$ affinity matrix, distortion measures were defined to measure the closeness of the data with the criteria that l neighbors of dissimilar data are dissimilar, while l neighbors of similar data are also similar. Then, in the calculated $n \times n$ affinity matrix, the NCut (Shi & Malik, 1997) method was applied to generate the final k clusters. Despite the fact that NSDR's clustering accuracy is high, scalability is not ensured since NSDR preserves the high complexity of the NCut method.

In (Fowlkes et al., 2004), p sample points/landmarks were selected randomly out of the initial n data points and eigendecomposition was performed to a highly reduced $p \times p$ submatrix. Then, the calculated p eigenvectors were used to estimate the original n eigenvectors based on the Nyström method (Nyström, 1930). In (Chen & Cai, 2011), the Landmark-based representation SC method (LSC) was proposed. By selecting p landmarks, a $n \times p$ affinity submatrix was created based on a sparse coding technique (Section 2.1), expressing the pairwise similarities between the p landmarks and the n data points. Two variations of LSC were proposed, LSC-R with the p landmarks being randomly selected; and LSC-K, where the preprocessing step of k-means is added into LSC, selecting thus p centroids as landmarks. In (Rafailidis et al., 2014), the Weighted PageRank algorithm was considered as a landmark selection strategy for LSC (LSC-WPR), outperforming LSC-R and LSC-K, by selecting more representative landmarks. However, all the accelerated methods work in an unsupervised way, by not exploiting pairwise constraints and thus preserving the accuracy of the unsupervised baseline methods, e.g. (Ng et al., 2002).

The implementation of NCut is publicly available.[6] By extending NCut we implemented the semi-supervised SC method of NSDR. Note that for the NSDR method, given a set $\mathcal{T}$ of $p = |\mathcal{T}|$ points in the pairwise constraints, the number of must-link and cannot-link constraints equals $(|\mathcal{M}| + |\mathcal{C}|)$. For the Nyström approximation based SC we choose the Matlab implementation with orthogonalization, which is publicly available.[7] The matlab codes of LSC-K and LSC-R are also publicly available[8] by the authors of

[5] All data sets were downloaded in the.mat format, publicly available at http://www.cad.zju.edu.cn/home/dengcai/Data/data.html.

[6] http://vision.ucsd.edu/~sagarwal/clustering.html.

[7] alumni.cs.ucsb.edu/~wychen/sc.html.

[8] http://www.cad.zju.edu.cn/home/dengcai/Data/Clustering.html.

Table 1
Data set description.

Data set	Size (n)	Dimensions (d)	Classes
COIL100	7200	1024	100
CMU PIE	11,554	1024	68
MNIST	70,000	784	10

Table 2
Methods comparison in terms of *Acc* (%).

	p = 5%	p = 10%	p = 15%	p = 20%
COIL100				
Accelerated				
SC-PC	**54.73 ± 1.47**	**66.32 ± 1.33**	**70.12 ± 0.92**	**75.78 ± 0.96**
Nyström	45.53 ± 2.10	46.18 ± 1.96	46.53 ± 1.89	46.92 ± 1.78
LSC-R	44.58 ± 2.07	49.88 ± 1.92	51.14 ± 1.78	52.80 ± 1.69
LSC-WPR	48.05 ± 1.12	53.83 ± 0.69	55.64 ± 0.88	59.36 ± 0.73
LSC-K	47.87 ± 1.23	52.20 ± 1.36	53.92 ± 1.25	55.59 ± 1.12
Baseline				
NCut	61.59 ± 1.29	61.41 ± 1.24	61.46 ± 1.12	61.09 ± 1.07
NSDR	62.31 ± 1.38	63.22 ± 1.46	65.98 ± 1.37	67.45 ± 1.28
CMU-PIE				
Accelerated				
SC-PC	26.87 ± 1.42	**41.60 ± 1.04**	**51.00 ± 0.89**	**56.99 ± 0.92**
Nyström	18.69 ± 2.43	20.56 ± 1.91	23.54 ± 2.07	25.51 ± 1.79
LSC-R	16.78 ± 1.99	17.61 ± 1.71	18.88 ± 1.69	20.07 ± 1.51
LSC-WPR	**27.58 ± 1.42**	28.01 ± 1.34	29.94 ± 0.95	30.34 ± 0.68
LSC-K	25.87 ± 1.54	26.01 ± 1.25	27.87 ± 1.33	28.51 ± 1.38
Baseline				
NCut	31.90 ± 1.24	32.09 ± 1.47	31.98 ± 1.36	32.07 ± 1.37
NSDR	6.82 ± 1.43	6.91 ± 1.38	6.22 ± 1.34	4.99 ± 1.41
	p = 300	p = 600	p = 900	p = 1200
MNIST				
Accelerated				
SC-PC	**75.21 ± 0.44**	**82.54 ± 0.68**	**85.85 ± 0.42**	**87.11 ± 0.31**
Nyström	47.94 ± 0.96	54.35 ± 0.97	54.72 ± 0.81	60.36 ± 0.86
LSC-R	60.63 ± 1.24	64.78 ± 1.07	65.02 ± 0.94	66.06 ± 0.77
LSC-WPR	71.20 ± 0.46	73.46 ± 0.54	76.24 ± 0.41	77.81 ± 0.34
LSC-K	70.54 ± 0.54	72.98 ± 0.63	74.54 ± 0.58	75.37 ± 0.45
Baseline				
NCut	N/A			
NSDR	N/A			

Bold values represent the highest scores.

(Chen & Cai, 2011). For LSC-WPR, we used the Gephi toolkit[9] to calculate the weighted PageRank values for the landmark selection step.

Regarding the parameter tuning, in k-means we set the default value of $t = 100$ iterations, for all methods. Following (Chen & Feng, 2012), for the NSDR method the l number of nearest neighbors is set to 5. For the proposed SC-PC method we also set l to 5, i.e. the l neighbors of the $|\mathcal{T}_a|$ points based on the affinity submatrix $\widehat{Z}$ (Section 2.3). Finally, following (Chen & Cai, 2011; Rafailidis et al., 2014) for LSC-K, LSC-R, LSC-WPR and SC-PC, we varied the number of r nearest landmarks (Section 2.1) from 2 to 10, where we concluded to 6, 4 and 3 for COIL100, CMU PIE and MNIST, respectively, with the exceptional case of $r = 3$ for LSC-K and LSC-R in CMU-PIE.

3.3. Results

In our experiments we varied the number of $p = |\mathcal{T}|$ points that participate in the pairwise constraints from 5% to 20% at a 5% step, expressed as a percentage of the n total size. The main reasons for limiting our p variation are (a) for the accelerated methods it must hold $p \ll n$ to preserve the computational cost low; and (b) in real-world applications it is easy to acquire raw data, while pairwise must-link and cannot-link constraints are expensive to generate (and thus to retrieve the respective

[9] http://gephi.github.io/.

Table 3
Methods comparison in terms of *NMI* (%).

	p = 5%	p = 10%	p = 15%	p = 20%
COIL100				
Accelerated				
SC-PC	73.58 ± 1.79	**80.71** ± **1.38**	**83.85** ± **1.08**	**86.76** ± **0.99**
Nyström	71.60 ± 2.14	72.14 ± 1.79	73.44 ± 1.64	73.59 ± 1.53
LSC-R	70.56 ± 1.98	72.66 ± 1.81	73.54 ± 1.72	74.61 ± 1.64
LSC-WPR	**74.79** ± **1.27**	78.21 ± 1.11	78.94 ± 0.97	80.31 ± 0.78
LSC-K	72.64 ± 1.42	76.18 ± 1.37	76.32 ± 1.38	76.45 ± 1.24
Baseline				
NCut	82.34 ± 1.54	82.37 ± 1.52	82.54 ± 1.44	82.23 ± 1.28
NSDR	83.40 ± 1.34	84.78 ± 1.48	85.74 ± 1.23	86.66 ± 1.35
CMU-PIE				
Accelerated				
SC-PC	32.26 ± 1.26	**45.35** ± **1.08**	**54.11** ± 1.03	**59.83** ± **0.97**
Nyström	36.49 ± 2.45	37.15 ± 1.81	40.80 ± 2.24	42.80 ± 1.76
LSC-R	25.51 ± 2.41	27.49 ± 2.29	28.02 ± 1.96	31.09 ± 1.74
LSC-WPR	**40.72** ± **1.47**	42.81 ± 1.17	44.89 ± 0.98	46.72 ± 1.07
LSC-K	32.63 ± 1.46	34.26 ± 1.38	35.27 ± 1.12	38.61 ± 1.11
Baseline				
NCut	49.50 ± 2.11	49.71 ± 2.41	49.66 ± 1.78	49.93 ± 1.66
NSDR	14.91 ± 1.95	14.93 ± 1.74	12.54 ± 1.82	8.79 ± 1.34
	p = 300	p = 600	p = 900	p = 1200
MNIST				
Accelerated				
SC-PC	62.58 ± 0.47	70.39 ± 0.51	73.50 ± 0.52	75.07 ± 0.43
Nyström	45.55 ± 1.03	48.30 ± 1.11	48.40 ± 0.82	50.76 ± 0.96
LSC-R	56.16 ± 1.14	61.92 ± 1.04	64.21 ± 0.94	65.26 ± 0.97
LSC-WPR	**70.23** ± **0.53**	**73.45** ± **0.47**	**74.69** ± **0.38**	**75.99** ± **0.41**
LSC-K	69.11 ± 0.51	72.37 ± 0.49	74.38 ± 0.53	75.55 ± 0.47
Baseline				
NCut	N/A			
NSDR	N/A			

Bold values represent the highest scores.

Table 4
Methods comparison in terms of computational cost (s).

	Accelerated					Baseline	
	SC-PC	Nyström	LSC-R	LSC-WPR	LSC-K	NCut	NSDR
COIL100							
p = 5%	5.6	6.46	**5.33**	6.19	6.16	429.41	456.33
p = 10%	6.61	10.5	**5.93**	6.76	8.1	429.41	460.08
p = 15%	8.16	28.4	**6.84**	7.67	9.51	429.41	463.39
p = 20%	10.43	56.92	**8.21**	9.03	11.84	429.41	466.84
CMU-PIE							
p = 5%	5.77	8.16	**5.28**	6.84	7.57	2023.93	2125.66
p = 10%	7.69	33.88	**6.45**	8.01	10.97	2023.93	2139.66
p = 15%	11.33	101.93	**7.48**	9.04	14.76	2023.93	2151.87
p = 20%	14.04	235.81	**8.56**	10.12	18.64	2023.93	2165.21
MNIST							
p = 300	9.58	9.76	**8.27**	22.13	14.05	N/A	N/A
p = 600	13.75	22.12	**9.96**	23.82	21.39	N/A	N/A
p = 900	17.02	41.03	**11.75**	25.61	29.03	N/A	N/A
p = 1,200	23.33	63.44	**16.78**	30.64	39.69	N/A	N/A

Bold values represent the highest scores.

p points that participate in the pairwise constraints). In the MNIST data set, to preserve the computational cost low we varied the p points from 300 to 1200, using a step of 300 points. Since in the NSDR and the proposed SC-PC methods, additional information is used by considering the set $\mathcal{T}$, with $p = |\mathcal{T}|$ of the points that participate in the pairwise constraints, in all experiments the set $\mathcal{T}$ is considered as training set, whereas the remaining set of n-p unconstrained points in the data set are considered as test set. To ensure fair comparison, for each method the same training/test sets were used. Following

Table 5
Methods Comparison in MNIST for $p = 1500$ and $p = 1800$.

	Acc (%)	*NMI* (%)	Comp. Cost (s)
p = 1500			
SC-PC	**88.14 ± 0.42**	**77.42 ± 0.39**	**27.47**
LSC-WPR	78.78 ± 0.44	76.54 ± 0.37	32.11
LSC-K	77.06 ± 0.52	76.69 ± 0.48	46.62
p = 1800			
SC-PC	**89.21 ± 0.49**	**79.33 ± 0.44**	**33.57**
LSC-WPR	81.48 ± 0.61	77.63 ± 0.54	36.19
LSC-K	80.72 ± 0.54	77.54 ± 0.47	56.35

Bold values represent the highest scores.

the evaluation protocol of (Chen & Cai, 2011; Chen & Feng, 2012), all experiments were repeated 10 times, where the means of (a) *Acc* (Cai, He, & Han, 2005), (b) Normalized Mutual Information (*NMI*) (Strehl & Ghosh, 2002) and (c) computational cost (in seconds) are reported. Additionally, for all experiments, we applied statistical pairwise t-tests, where the calculated differences of means between the runs were insignificant at 0.05 level. The direct comparison of our proposed method with the other accelerated methods in terms of Accuracy, *NMI* and computational cost are presented in Tables 2–4 and the best score is highlighted. Moreover, in the same tables the baseline methods of NCut and NSDR are presented for coherency. It should be noted, however, that the baseline methods are not directly compared to the accelerated ones due their computational complexity (they require significant larger processing times), while they do not always outperform the latter methods in terms of accuracy and *NMI*.

All experiments were performed on a Windows 7 PC with Intel core i5-2430M CPU @ 2.4 GHz with 8 GB RAM, using Matlab 2010a.

3.4. Discussion

With respect to the clustering accuracy, the proposed SC-PC method outperforms the competitive accelerated methods. By dynamically adapting and propagating the constraints to the local neighborhood, SC-PC clearly has higher *Acc* and *NMI* than the accelerated methods of Nyström and LSC-R on the three evaluation data sets. Compared to LSC-K, the proposed SC-PC method achieves higher clustering accuracy in terms of *Acc* and *NMI* in all datasets except for the MNIST data set. Even in this case, SC-PC's *NMI* performance is comparable to LSC-K, while the computational cost is significantly reduced (Table 4). Moreover, SC-PC achieves higher *Acc* and *NMI* than LSC-K in COIL100 and CMU PIE. The proposed SC-PC also outperforms LSC-WPR in most cases. However, for a small number of landmarks e.g. $p = 5\%$ in COIL100 and CMU PIE, as wel as for all p variations in MNIST, with $p \leq 1.71\%$ (where $p = 300$, 600, 900 and 1200 correspond to $p = 0.42\%$, 0.85%, 1.28% and 1.71%), LSC-WPR achieves higher *NMI* score than SC-PC, whereas for larger p values ($>5\%$) SC-PC is more accurate than LSC-WPR in COIL100 and CMU-PIE. This happens because for a small number of landmarks, LSC-WPR identifies more important landmarks based on the weighted PageRank algorithm; however, by increasing the number of landmarks, LSC-WPR considers less important data points as landmarks, limiting thus the clustering accuracy, compared to SC-PC. To verify this, we conducted the following experiment; we increased the number of the p landmarks in the MNIST data set, considering $p = 1500$ and $p = 1800$ landmarks, corresponding to $p = 2.14\%$ and $p = 2.57\%$, respectively. As presented in Table 5, for both p variations SC-PC achieves higher *Acc* and *NMI* in less computational time compared to LSC-WPR and LSC-K. Finally, a very interesting finding of our experiments is the fact that our approach outperforms NSDR even though it uses its base concept of must-link and cannot-link constraints. This fact is credited to the sparse coding approach that filters out possible outliers and noise that could drive the clustering in poor results as well as the dynamic adaptation of the local neighborhoods to the characteristics of each connected component.

With respect to the computational time, LSC-R has the lowest computational cost, outperforming the Nyström method, as it was also experimentally shown in (Chen & Cai, 2011). The proposed SC-PC method preserves the processing cost relatively low, compared to the rest of accelerated methods of LSC-K and LSC-WPR, while SC-PC achieves higher *Acc* and *NMI* by exploiting the pairwise constraints. Especially for the large-scale data set of MNIST, by varying the number of the p constrained points in the range of 300–1800, SC-PC needs 9.58–33.57 s, which is almost two times faster than LSC-K which needs 14.05–56.35 s. Also, SC-PC is faster than LSC-WPR which requires 22.13–36.19 s. Compared to NCut and the semi-supervised NDSR method, it is clear that the proposed SC-PC method is at least 40 times faster with cases where the difference reaches to 350 times faster. At this point we must mention that in addition to the high processing cost, NCut and NSDR cause memory "overflows" in the large-scale data set of MNIST, making them inappropriate for larger datasets.

Summarizing our results, the proposed SC-PC method outperforms the accelerated methods in terms of clustering accuracy while it manages to keep computational costs relatively close to the fast accelerated method of LSC-R, compared to LSC-K and LSC-WPR. Additionally, SC-PC outperforms, in most of the cases, the clustering accuracy of the semi-supervised baseline NSDR method, while it drastically reduces the processing cost.

4. Conclusion

In this paper we present an efficient method for accurate and scalable SC. In particular, we handle real-world problems which either decrease the clustering accuracy or significantly increase the computation time. This happens because state-of-the-art methods of spectral clustering either follow unsupervised strategies or lack scalability. The proposed SC-PC method achieves high clustering accuracy and ensures scalability, by (a) extracting and setting the pairwise constraints to a small affinity submatrix; (b) computing connected components to dynamically adapt the local neighborhoods of points of the same component; (c) performing constraints propagation to the adapted local neighborhoods; and (e) following a landmark spectral clustering strategy. As it was experimentally shown, the proposed method outperforms state-of-the-art spectral clustering methods that either follow semi-supervised or scalable strategies, in terms of clustering accuracy and computational cost.

Moreover, in real-world applications continuous updates are required as the data sets evolve over time. Recently, several incremental strategies (Dhanjal, Gaudel, & Clémençon, 2011; Ning, Xu, Chi, Gong, & Huang, 2010) have been proposed in the literature, by efficiently updating the eigenspace. (Chi, Song, Zhou, Hino, & Tseng, 2009; Xu, Kliger, & Hero, 2010, 2014) able to handle not only insertion/deletion of data points but also similarity changes between existing points. In our future research we plan to examine both the incremental and evolving strategies of the proposed SC-PC method in the context of spectral clustering in Big Data (Mall, Langone, & Suykens, 2013).

Acknowledgment

This work was supported by the EC FP7 funded Project CUBRIK, ICT-287704 (www.cubrikproject.eu).

References

Bsat, M., Baker, S., & Sim, T. (2001). The CMU pose, illumination, and expression (PIE) database of human faces. In *CMU Robotics Institute*.

Cai, D., He, X., & Han, J. (2005). Document clustering using locality preserving indexing. *IEEE Transactions on Knowledge and Data Engineering, 17*(12), 1624–1637.

Cao, J.-Z., Chen, P., Dai, Q., & Ling, B. W.-K. (2014). Local information-based fast approximate spectral clustering. *Pattern Recognition Letters, 38*, 63–69.

Cevikalp, H., & Triggs, B. (2010). Face recognition based on image sets. In *CVPR* (pp. 2567–2573). IEEE.

Chan, P. K., Schlag, M. D. F., & Zien, J. Y. (1993). Spectral K-way ratio-cut partitioning and clustering. In *DAC* (pp. 749–754).

Chang, H., & Yeung, D. Y. (2008). Robust path-based spectral clustering. *Pattern Recognition, 41*(1), 191–203.

Chen, X., & Cai, D. (2011). Large scale spectral clustering with landmark-based representation. In W. Burgard & D. Roth (Eds.), *AAAI*. AAAI Press.

Chen, W., & Feng, G. (2012). Spectral clustering: A semi-supervised approach. *Neurocomputing, 77*(1), 229–242.

Chen, W.-Y., Song, Y., Bai, H., Lin, C.-J., & Chang, E. Y. (2011). Parallel spectral clustering in distributed systems. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 33*(3), 568–586.

Chi, Y., Song, X., Zhou, D., Hino, K., & Tseng, B. L. (2009). On evolutionary spectral clustering. *TKDD, 3*(4).

Chung (1997). Spectral graph theory (reprinted with corrections). In *CBMS: Conference board of the mathematical sciences, regional conference series*.

Cun, Y. L. L., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of IEEE, 86*(11), 2278–2324.

Dhanjal, C., Gaudel, R., & Clémençon, S. (2011). Incremental spectral clustering with the normalised laplacian.

Filippone, M., Camastra, F., Masulli, F., & Rovetta, S. (2008). A survey of kernel and spectral methods for clustering.

Fowlkes, C., Belongie, S., Chung, F. R. K., & Malik, J. (2004). Spectral grouping using the nyström method. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 26*(2), 214–225.

Härdle, W. (1992). *Applied nonparametric regression*. Berlin: Springer Verlag.

Huang, H.-C., Chuang, Y.-Y., & Chen, C.-S. (2012). Affinity aggregation for spectral clustering. In *CVPR* (pp. 773–780). IEEE.

Iso, K. (2010). Speaker clustering using vector quantization and spectral clustering. In *ICASSP* (pp. 4986–4989). IEEE.

Kang, U., Meeder, B., Papalexakis, E. E., & Faloutsos, C. (2014). HEigen: Spectral analysis for billion-scale graphs. *IEEE Transactions on Knowledge and Data Engineering, 26*(2), 350–362.

Kulis, B., Basu, S., Dhillon, I. S., & Mooney, R. J. (2009). Semi-supervised graph clustering: A kernel approach. *Machine Learning, 74*(1), 1–22.

Liu, J., Wang, C., Danilevsky, M., & Han, J. (2013). Large-scale spectral clustering on graphs. In F. Rossi (Ed.), *IJCAI*. IJCAI/AAAI.

Luxburg, U. V. (2007). A tutorial on spectral clustering. *Statistics and Computing, 17*(4), 395–416.

Mall, R., Langone, R., & Suykens, J. A. K. (2013). Kernel spectral clustering for big data networks. *Entropy, 15*(5), 1567–1586.

Nene, S. A., Nayar, S. K., & Murase, H. (1996). Columbia object image library (COIL-100). In *Columbia University*.

Ng, A. Y., Jordan, M., & Weiss, Y. (2002). On spectral clustering: Analysis and an algorithm. *Advances in neural information processing systems* (Vol. 14). Cambridge, MA: MIT Press.

Ning, H. Z., Xu, W., Chi, Y., Gong, Y. H., & Huang, T. S. (2010). Incremental spectral clustering by efficiently updating the eigen-system. *Pattern Recognition, 43*(1), 113–127.

Nyström, E. J. (1930). Über die praktische Auflösung von Integralgleichungen mit Anwendungen auf Randwertaufgaben. *Acta Mathematica, 54*, 185–204.

Paccanaro, A., Chennubhotla, C., Casbon, J., & Saqi, M. (2003). Spectral clustering of protein sequences. In *Proceedings of the international joint conference on neural networks, 2003* (Vol. 4, pp. 3083–3088).

Rafailidis D., Constantinou E., & Y. Manolopoulos (2014). Scalable spectral clustering with weighted PageRank. In *Proc. of 4th int. conf. on model and data engineering* (pp. 289–300).

Shi, J., & Malik, J. (1997). Normalized cuts and image segmentation. In *IEEE conf. computer vision and pattern recognition*.

Strehl, A., & Ghosh, J. (2002). Cluster ensembles — A knowledge reuse framework for combining multiple partitions. *Journal of Machine Learning Research, 3*, 583–617.

Tarjan, R. E. (1972). Depth first search and linear graph algorithms. *Journal of Computing, 1*(2), 146–160.

Tatsuma, A., & Aono, M. (2009). Multi-fourier spectra descriptor and augmentation with spectral clustering for 3D shape retrieval. *The Visual Computer, 25*(8). xx–yy.

Tung, F., Wong, A., & Clausi, D. A. (2010). Enabling scalable spectral clustering for image segmentation. *Pattern Recognition, 43*(12), 4069–4076.

Wagstaff, K., Cardie, C., Rogers, S., & Schroedl, S. (2001). Constrained K-means clustering with background knowledge. In *Proc. 18th international conf. on machine learning* (pp. 577–584). San Francisco, CA: Morgan Kaufmann.

Xu, K. S., Kliger, M., & Hero, A. O. III, (2010). Evolutionary spectral clustering with adaptive forgetting factor. In *ICASSP* (pp. 2174–2177). IEEE.

Xu, K. S., Kliger, M., & Hero, A. O. III, (2014). Adaptive evolutionary clustering. *Data Mining and Knowledge Discovery, 28*(2), 304–336.

Yan, D., Huang, L., & Jordan, M. I. (2009). Fast approximate spectral clustering. Technical report.