sciendo

Int. J. Appl. Math. Comput. Sci., 2019, Vol. 29, No. 1, 125–137
DOI: 10.2478/amcs-2019-0010

amcs

CONSTRAINED SPECTRAL CLUSTERING VIA MULTI–LAYER GRAPH EMBEDDINGS ON A GRASSMANN MANIFOLD

ALEKSANDAR TROKICIĆ [a,*], BRANIMIR TODOROVIĆ [a]

[a] Department of Computer Science
University of Niš, Višegradska 33, 18000 Niš, Serbia
e-mail: {aleksandar.trokicic,branimirtodorovic}@pmf.ni.ac.rs

We present two algorithms in which constrained spectral clustering is implemented as unconstrained spectral clustering on a multi-layer graph where constraints are represented as graph layers. By using the Nystrom approximation in one of the algorithms, we obtain time and memory complexities which are linear in the number of data points regardless of the number of constraints. Our algorithms achieve superior or comparative accuracy on real world data sets, compared with the existing state-of-the-art solutions. However, the complexity of these algorithms is squared with the number of vertices, while our technique, based on the Nyström approximation method, has linear time complexity. The proposed algorithms efficiently use both soft and hard constraints since the time complexity of the algorithms does not depend on the size of the set of constraints.

Keywords: spectral clustering, constrained clustering, multi-layer graph, Grassmann manifold, Nyström method, Laplacian matrix.

1. Introduction

Given a set of data points and a definition of similarity, the clustering algorithm divides data into groups, with high similarity between the elements of the same group and dissimilarity between the elements of different groups. Many clustering methods are based on explicit or implicit assumptions that clusters form convex regions in Euclidean space; however, we will consider here another approach, called spectral clustering (Ng *et al.*, 2002; Von Luxburg, 2007; White and Smyth, 2005). It uses the spectrum (eigenvalues and eigenvectors) of the similarity (affinity) matrix, the elements of which represent similarity between data points, to perform the clustering as similarity graph partitioning. Unlike the Euclidean space based clustering algorithms such as k-means, spectral clustering solutions can separate arbitrarily shaped clusters.

However, spectral clustering algorithms are not widely accepted as competitors to hierarchical clustering or k-means for large-scale data mining problems. The obvious reason is that spectral clustering algorithms compute eigenvectors of the affinity matrix, and in general this operation has cubic complexity with respect to the number of data points. Another issue that should be considered is how to enable the user to guide the clustering process and to incorporate prior domain knowledge in order to improve the accuracy of the final clustering. Constrained clustering algorithms (Basu *et al.*, 2008) incorporate domain knowledge in the form of constraints that specify the cluster membership of sets of instances. Algorithms often use both must-link constraints, which specify sets of instances that belong to the same cluster, and cannot-link constraints, which specify instances that belong to different clusters.

Constrained spectral clustering algorithms from previous studies fall into three different categories. In the first category, they are introduced by changing the adjacency matrix (Kamvar *et al.*, 2003; Xu *et al.*, 2005; Lu and Carreira-Perpinan, 2008). In the second one, constraints are introduced by bounding the feasible solution space (Coleman *et al.*, 2008; De Bie *et al.*, 2004; Li *et al.*, 2009). In the third category, there are algorithms that modify the problem into an constrained optimization problem (Wang *et al.*, 2014). The approach proposed in this paper consists of two main steps. We transform the input into a multi-layer graph. The latter is a structure

*Corresponding author

which represents a set of graphs with a common vertex set. Each graph in the structure is called a layer. The first layer is a similarity graph and the other layers are constrained graphs formed from a subset of the set of constraints. The second step is unconstrained spectral clustering on the newly formed multi-layer graph. For spectral clustering on multi-layer graphs we use the algorithm described by Dong *et al.* (2014). They used an algorithm that belongs to a class of subspace-based methods (Turk and Pentland, 1991), based on graph layer embedding onto the Grassmann manifold. The previous work that uses subspace analysis on Grassmann manifolds includes that by Hamm and Lee (2008; 2009) as well as Harandi *et al.* (2011).

The complexity of these constrained spectral clustering algorithms as well as that of the ordinary spectral clustering algorithm is $O(n^2 + t_{km})$, where t_{km} stands for the number of iterations of a k-means algorithm and n represents the number of data points. In order to obtain better time complexity, we applied the Nyström approximation. In the literature there are several approaches (Choromanska *et al.*, 2013; Fowlkes *et al.*, 2004; Li *et al.*, 2011) which incorporate the Nyström method into spectral clustering. However, only Choromanska *et al.* (2013) provided performance guarantees for their algorithm. By incorporating this solution into our constrained spectral clustering algorithm we get one with complexity $O(nk^2 + t_{km})$, where $k \ll n$ is the number of selected data points.

Li *et al.* (2015) proposed a fast constrained spectral clustering algorithm, with time complexity linear in n. However, they assume that the number of constraints c is small, $c \ll n$, which in general can be as large as n^2. The algorithm by Li *et al.* (2015) has either time complexity squared in c or memory complexity squared in n depending on the implementation, which is intractable for large n and c. On the other hand, our algorithm has time and memory complexity linear in n regardless of the number of constraints. To our knowledge, it is the only constrained spectral clustering algorithm that uses both hard and soft constraints and achieves this complexity regardless of the size of the set of constraints.

We make the following contributions:

- We show that the constrained spectral clustering algorithm can be implemented as unconstrained spectral clustering on a multi-layer graph where constraints are represented as layers in that graph. The time complexity of this algorithm is $\mathcal{O}(n^2)$.
- In order to solve the time complexity problem, in Section 4 we present an algorithm based on a Nystrom method with linear time and memory complexity.

The paper is structured as follows. In Section 2, we review spectral clustering algorithms. In Section 3, we introduce constrained spectral clustering as unconstrained spectral clustering on multi-layer graphs. The fast constrained spectral algorithm that uses the Nyström method is described in Section 4. Finally, experimental results are described in Section 5, and the conclusion is given in the final section.

Algorithm 1. Spectral clustering (Ng *et al.*, 2002).

Input: W and k;
Compute D and L;
Compute U from L, normalize it and then cluster its rows;
Output: Clusters $C_1, \ldots, C_k$

2. Spectral clustering

In this section we will briefly explain the spectral clustering algorithm on a graph G. Let $G = (V, W)$ be an undirected weighted graph, with the set of n vertices $V = \{v_1, v_2, \ldots, v_n\}$ and an adjacency matrix W. Specifically, W is a symmetric matrix where positive real number W_{ij} is the weight of edge (v_i, v_j). If $W_{ij} = 0$, then there is no edge between v_i and v_j. The diagonal matrix D such that $D_{ii} = \sum_{j=1}^{n} W_{ij}$ is called the degree matrix and D_{ii} the degree of vertex v_i.

Given an undirected similarity graph $G = (V, W)$ and a number of desired clusters k, we wish to partition graph vertices into k groups. In particular, a clustering algorithm goal is to find a partition that minimizes the normalized cut. Specifically, for a given vertex subset $A \subset V$, $cut(A, V \setminus A)$ is the sum of edge weights W_{ij}, where $v_i \in V$ and $v_j \in V \setminus A$. In addition, $vol(A)$ represents the sum of degrees of vertices in A. Therefore, a normalized cut of a cluster (partition) set $(A_1, A_2, \ldots, A_k)$ is the sum

$$NCut(A_{1:k}) = \sum_{i=1}^{i=k} \frac{cut(A_i, V \setminus A_i)}{vol(A_i)}.$$

We call a set of clusters $A_{1:k}$, such that $NCut(A_{1:k})$ reaches a minimum, a solution to the minNCut problem. According to Von Luxburg (2007), relaxing this problem yields

$$\min_{U \in \mathbb{R}^{n \times k}} \operatorname{tr}(U^T L U) \quad \text{such that} \quad U^T U = I, \quad (1)$$

where $L = D^{-\frac{1}{2}}(D - W)D^{-\frac{1}{2}}$ is a normalized graph Laplacian, with the following properties.

Property 1. (*Von Luxburg, 2007*) For the normalized graph Laplacian L, the following statements are true:

- L is positive semi-definite matrix;
- L has n non-negative real-valued eigenvalues $0 = \lambda_1 \leq \cdots \leq \lambda_n$,

- 0 is an eigenvalue of L with an eigenvector $D^{\frac{1}{2}}\mathbf{1}$.

The solution to the problem (1) is a matrix U, called a *spectral embedding matrix*, the columns of which are first k eigenvectors of Laplacian L. Cluster assignments are derived as a solution to k-means clustering on rows of U.

Furthermore, the column space of matrix U will be used as a subspace representation of a graph for clustering on a multi-layer graph as in the work of Dong *et al.* (2014), i.e., as a point in the Grassmann manifold. Now we will briefly describe the latter and its basic properties.

2.1. Grassmann manifold.

Definition 1. (*Hamm and Lee, 2008*) The *Grassmann manifold* $\mathcal{G}(k,n)$ is the set of k-dimensional linear subspaces of $\mathbb{R}^n$.

An orthonormal matrix $Y \in \mathbb{R}^{n\times k}$ can be used for representation of a point, column space $span(Y)$, in $\mathcal{G}(k,n)$. This kind of representation is not unique, because $span(Y_1) = span(Y_2)$ can be true for two different matrices Y_1, Y_2. Therefore when we say that matrix Y belongs to $\mathcal{G}(k,n)$, we mean that its column space belongs to $\mathcal{G}(k,n)$. The distance between two elements of $\mathcal{G}(k,n)$ can be defined using principal angles (Golub and Van Loan, 1996).

Definition 2. Let Y_1 and Y_2 be two orthonormal matrices from $\mathbb{R}^{n\times k}$. The *principal angles* $0 \le \theta_1 \le \cdots \le \theta_k \le \pi/2$ between two subspaces $span(Y_1)$ and $span(Y_2)$ are defined by

$$\cos\theta_i = \max_{u_i \in span(Y_1)} \max_{v_i \in span(Y_2)} u_i^T v_i$$

subject to

$$u_i^T u_i = 1, \quad v_i^T v_i = 1, \quad u_j^T u_i = 0,$$

$$v_j^T v_i = 0, \qquad j \in \{1,\dots,i-1\}.$$

Note that θ_1 represents the smallest angle between two unit vectors chosen from $span(Y_1)$ and $span(Y_2)$, respectively. Angle θ_2 is similarly chosen as the smallest angle between two unit vectors chosen from $span(Y_1)$ and $span(Y_2)$, respectively, with one additional condition that these vectors should be orthogonal to the previously chosen vectors. The remaining angles are chosen similarly. The principal angles can be computed from the singular value decomposition (SVD) of $Y_1^T Y_2 = U\cos\Theta V^T$, where $U = [u_1 \dots u_k]$, $V = [v_1 \dots v_k]$, and $\cos\Theta$ is the diagonal matrix $\cos\Theta = diag(\cos\theta_1,\dots,\cos\theta_k)$. According to Hamm and Lee (2008) the projection distance is defined as $d_P^2(Y_1,Y_2) = (\sum_{i=1}^k \sin^2\theta_i)$, which is, according to Dong *et al.* (2014), equivalent to

$$\begin{aligned} &d_P^2(Y_1,Y_2) \\ &= \frac{1}{2}\operatorname{tr}(Y_1^T Y_1 + Y_2^T Y_2 - 2Y_1 Y_1^T Y_2 Y_2^T) \\ &= \frac{1}{2}||Y_1 Y_1^T - Y_2 Y_2^T||_F^2. \end{aligned} \tag{2}$$

Algorithm 2. Spectral clustering on a multi-layer graph (Dong *et al.*, 2014).

Input: m, $W_1,\dots,W_m$ and k;
For $(\forall i)$ compute L_i and U_i;
Compute L_{mod};
Compute matrix U, normalize it and cluster its rows;
Output: Clusters $C_1,\dots,C_k$

2.2. Spectral clustering on multi-layer graphs.

Now we will briefly explain a spectral clustering algorithm on multi-layer graphs developed by Dong *et al.* (2014). This algorithm will be used in our solution to the constrained spectral clustering problem. A multi-layer graph with m layers is a set of m individual graphs $M_G = \{(G_i = (V,E_i,W_i))_{i\le m}\}$ with the same set of vertices. V is the common set of vertices, E_i is the set of edges, W_i is the adjacency matrix of a layer i. For each layer, graph Laplacian L_i and spectral embedding matrix U_i are computed. Dong *et al.* (2014) proposed to find a matrix U such that $span(U)$ is as close as possible to $\{span(U_i)\}_{1\le i\le m}$, and such that U preserves local distances of each layer and to perform k-means clustering on rows of U. Furthermore, in order to compute a matrix U, they proposed the following optimization problem for clustering on a multi-layer graph:

$$\begin{aligned} \min_{\substack{U\in\mathbb{R}^{n\times k}\\ U^TU=I}} &\sum_{i=1}^m \operatorname{tr}(U^T L_i U) \\ &+ \alpha[km - \sum_{i=1}^m \operatorname{tr}(UU^T U_i U_i^T)], \end{aligned} \tag{3}$$

where α is a regularization parameter. This problem is equivalent to

$$\min_{\substack{U\in\mathbb{R}^{n\times k}\\ U^TU=I}} \operatorname{tr}(U^T(\sum_{i=1}^m L_i - \alpha\sum_{i=1}^m U_i U_i^T)U). \tag{4}$$

Therefore, the columns of a solution matrix U are the first k eigenvectors of a modified Laplacian matrix,

$$L_{mod} = \sum_{i=1}^m L_i - \alpha\sum_{i=1}^m U_i U_i^T. \tag{5}$$

Note that the order of layers is irrelevant to the derivation of the matrix L_{mod}, and therefore it is irrelevant to the clustering.

3. Constrained spectral clustering

In this section, we will show our algorithm for constrained spectral clustering. We are given a similarity graph $G = (V, W)$ derived from input feature vectors and a set of constraints $\mathcal{PC}$. Elements of a set $\mathbf{PC}$ have the following form $e = \{v_i, v_j, t, type\}$:

1. Pairwise constraint e refers to the vertices v_i and v_j.
2. $type \in \{ML, CL\}$ refers to the type of constraint, i.e., whether it is a must link or a cannot link.
3. Number t is a weight of constraints. If for a weight t of every constraint from a set $\mathcal{PC}$ we have $t \in \{0, 1\}$, then the constraints are called *hard* and, if for some constraints the weight is $t \in (0, 1)$, then the constraints are called *soft*.

There are several methods in the literature for solving this problem. Here we will briefly mention two that give the best results according to Wang *et al.* (2014). Kamvar *et al.* (2003) developed an algorithm that can be applied only to a set of hard constraints, called the spectral learning algorithm, by changing the adjacency matrix:

1. If $\{v_i, v_j, 1, ML\} \in \mathcal{PC}$, then the edge weight $W(i, j)$ is set to 1,
2. If $\{v_i, v_j, 1, CL\} \in \mathcal{PC}$, then the edge weight $W(i, j)$ is set to 0.

After that spectral clustering is applied. A matrix whose columns are largest k eigenvectors of $(W + d_{\max}I - D)/d_{\max}$, where D is a diagonal degree matrix and $d_{\max}$ the largest degree, is formed, and a k-means algorithm is performed on rows of this matrix.

Wang *et al.* (2014) have a different approach and their algorithm can be applied to both a data set with hard constraints and a data set with soft constraints. Their idea was to find the cluster indicator vector as a solution to a constrained optimization problem. We will first explain their approach using a case with two clusters. First they create a constraint matrix Q in the following way:

$$Q(i,j) = \begin{cases} 1, & \text{if } \{v_i, v_j, 1, ML\} \in \mathcal{PC}, \\ -1, & \text{if } \{v_i, v_j, 1, CL\} \in \mathcal{PC}, \\ 0, & \text{otherwise.} \end{cases} \quad (6)$$

In the case of soft constraints instead of values $\{-1, 1\}$ in the constraint matrix, they put weight of constraints $\{-t, t\}$. The constraints matrix is normalized in the same way as graph Laplacian $Q \leftarrow D^{-\frac{1}{2}}QD^{-\frac{1}{2}}$. Wang *et al.* (2014) defined the constrained spectral clustering problem for two partitions as the following optimization one, whose the solution is the cluster indicator vector v:

$$\min_{v \in \mathbb{R}^n} v^T Lv$$

such that

$$v^T Qv \geq \alpha, \quad ||v||^2 = vol, \quad v \neq D^{\frac{1}{2}}\mathbf{1}, \quad (7)$$

where the graph volume vol is the sum of degrees of its vertices $vol = \sum_{i=1}^{n} D_{ii}$. For a chosen $\beta < \alpha$, it can be proven that a solution v to a constrained spectral clustering problem (7) is a solution to the following generalized eigenvalue problem:

$$Lv = \lambda\Big(Q - \frac{\beta}{vol}I\Big)v. \quad (8)$$

In addition, according to Wang *et al.* (2014), only generalized eigenvectors associated with positive eigenvalues can be the solution to the problem (7). Therefore, the algorithm by Wang *et al.* (2014) can be summarized as follows:

1. Solve the generalized eigenvalue problem (8).
2. Create a feasible set of generalized eigenvectors v associated with positive eigenvalues and normalize them so that $v^T v = vol$.
3. From among the feasible set, choose vector $v^* = \arg\min_v v^T Lv$.
4. Based on the cluster indicator vector $u^* = D^{-\frac{1}{2}}v^*$, construct clusters.

In the case of k clusters, in Step 3, the matrix $V^* = \arg\min_{V \in \mathbb{R}^{n \times (k-1)}} \operatorname{tr}(V^T LV)$ is constructed so that its columns are chosen from the feasible set. In Step 4, k-means clustering is performed on the rows of the matrix $D^{-\frac{1}{2}}V^*$.

We propose to transform the input similarity graph $G = (W, E)$ and the set of constraints $\mathcal{PC}$ into the multi-layer graph M_G. In the light of this transformation, we implement a constrained spectral clustering algorithm on the input similarity graph and the set of constraints as an unconstrained spectral clustering algorithm on the multi-layer graph M_G. First, we will construct a multi-layer graph that will correspond to our constrained problem. Every layer of this multi layer graph will have the same vertex set V. Obviously, we use the input similarity graph as a first layer. Therefore its adjacency matrix is equal to the input graph adjacency matrix $W_1 = W$. The next step is to embed constraints into following layers. We propose to use two layers as embedded constraints; one for must-link constraints and the other for cannot link constraints. Therefore, we construct a 3-layer graph in which the first layer is the input graph and the second and third layers represent constraints.

We will encode must-link constraints with the following adjacency matrix W_2 of the second layer:

$$W_2(i,j) = \begin{cases} 1 & \text{if } \{v_i, v_j, 1, ML\} \in \mathcal{PC}, \\ 0 & \text{otherwise,} \end{cases} \quad (9)$$

Algorithm 3. Constrained spectral clustering as clustering on a multi layer graph (CSP-ML).

Input: W, k and $\mathcal{PC}$;
Create layers of multi-layer graph (W_1, W_2 and W_3);
For each i compute L_i and spectral embedding U_i;
Compute L_{mod};
Compute U, normalize it and cluster its rows;
Output: Clusters $C_1, \ldots, C_k$

and cannot link constraints with the following adjacency matrix W_3 of the third layer:

$$W_3(i,j) = \begin{cases} 0 & \text{if } \{v_i, v_j, 1, CL\} \in \mathcal{PC}, \\ 1 & \text{otherwise.} \end{cases} \quad (10)$$

In the final step, we cluster this multi-layer graph using Algorithm 2. Our procedure is summarized as Algorithm 3. Note that the order of layers is irrelevant to the algorithm.

We draw several conclusions regarding this algorithm:

1. This algorithm is not dependent on hard constraints. For example, let $c = \{v_i, v_j, t_c, type\} \in \mathcal{PC}$ be an input constraint. The weight of the constraint is t_c, and therefore we say that the edge weight between vertices v_i and v_j in the constraint layer is t_c.

2. Since constraints are encoded with two graph layers with n vertices, the complexity of this algorithm will only depend on n and not on the number of constraints. Therefore, we can add as many constraints as we want and this will not influence the speed of the algorithm.

3. As we add more constraints, the algorithm will converge towards the true partition. When we add all constraints, the second and third layers will represent the true partition and they will have significant influence on the clustering.

4. Fast constrained spectral clustering

The complexity of Algorithm 3 is $\mathcal{O}(kn^2 + t_{km})$. Therefore, its application to large scale graphs will be slow. In order to solve this problem, the complexity of spectral embedding needs to be reduced. We will apply here a similar idea as in fast spectral clustering via the Nyström method, as implemented by Choromanska *et al.* (2013), in order to reduce the complexity of our spectral clustering algorithm. We will sample a set of l columns $\mathcal{L}$, common for each layer's adjacency matrix, and use it to approximate graph Laplacian C_i and from it compute approximate spectral embedding U_i for each layer. Next, we will approximate the modified graph Laplacian with $\tilde{C}_{mod} = \sum_{i=1}^{m} C_i + \alpha \sum_{i=1}^{m} U_i U_i^T(:, \mathcal{L})$ and use its first k-eigenvectors for k-means clustering.

Now we will briefly review a Nyström method for approximation of a symmetric positive semi-definite matrix $X \in \mathbb{R}^{n \times n}$. Matrix C is derived from matrix X by sampling its columns. In particular, let $\mathcal{L}$ be the set of randomly chosen column indices. Hence, $C = X(\mathcal{L}, :)$ and its dimension is $l \times n$. Matrix Q is derived from C by choosing l rows in the following way: $Q = C(:, \mathcal{L})$. Thus we can compute the eigenvalue decomposition of a matrix $Q = U_Q \Sigma_Q U_Q^T$ since it is a symmetric matrix. The Nystrom method produces an approximation $\tilde{X}$ of the matrix X with the following eigenvalue decomposition:

$$\tilde{X} = \Big(\sqrt{\frac{l}{n}} C U_Q \Sigma_Q^+\Big)\Big(\frac{n}{l}\Sigma_Q\Big)\Big(\sqrt{\frac{l}{n}} C U_Q \Sigma_Q^+\Big)^T. \quad (11)$$

The complexity of this method is $\mathcal{O}(nl^2)$. The next theorem, by Kumar *et al.* (2009), gives an upper bound on the error of the Nyström method.

Theorem 1. (Kumar *et al.*, 2009) *Let $K \in \mathbb{R}^{n \times n}$ be a symmetric positive semi-definite matrix. Assume that l columns of K are sampled uniformly at random without replacement, $\tilde{K}_r$ is the r-rank Nyström approximation and K_r is the best r-rank approximation of K. Let*

$$\delta \in (0,1), \quad \epsilon > 0, \quad l \geq \frac{64r}{\epsilon^4},$$

$$\eta = \sqrt{\frac{\log(\frac{2}{\delta})\xi(l, n-l)}{l}},$$

where

$$\xi(a,b) = \frac{ab}{a+b-1/2}\frac{1}{1-\frac{1}{2\max(a,b)}}.$$

If $\gamma = \sum_{i \in D(l)} K_{ii}$, where $D(l)$ is the set of indices of l largest diagonal elements of K, and if $\theta = \sqrt{n \sum_{i=1}^{n} K_{ii}^2}$, then with probability $1-\delta$ the following expression is true:

$$||K - \tilde{K}_r||_F \leq ||K - K_r||_F + \epsilon\Big[\Big(\frac{n}{l}\gamma\Big)\Big(\theta + \eta \max_{i=\overline{1,n}} nK_{ii}\Big)\Big]^{\frac{1}{2}}. \quad (12)$$

Now we will describe the algorithm for fast spectral clustering of a graph $G = (V, W)$ via the Nyström method from the work of Choromanska *et al.* (2013). Let $\mathcal{L}$ be the set of indices of l sampled columns of adjacency matrix W. Matrix $\hat{W} = W(:, \mathcal{L})$ is an adjacency matrix with sampled columns. Now two sparse diagonal degree matrices are computed:

$$D \in \mathbb{R}^{n \times n} \quad \text{such that} \quad D_{ii} = \frac{1}{\sqrt{\sum_{j=1}^{l} \hat{W}_{ij}}}, \quad (13)$$

$$\Delta \in \mathbb{R}^{l\times l} \quad \text{such that} \quad \Delta_{jj} = \frac{1}{\sqrt{\sum_{i=1}^{n} \hat{W}_{ij}}}. \tag{14}$$

The following matrix is an approximation of the sampled graph Laplacian:

$$C = I(:, \mathcal{L}) - \sqrt{\frac{l}{n}} DW\Delta.$$

Now matrix Q is computed as $sym(C(\mathcal{L}, :))$. In the next step, eigenvalue decomposition of matrix Q is computed: $Q = U_Q \Sigma_Q U_Q^T$. The diagonal eigenvalue matrix of the approximate graph Laplacian matrix is $\tilde{\Sigma} = \frac{n}{l}\Sigma_Q$, and the approximate eigenvector matrix is

$$\tilde{U} = \sqrt{\frac{l}{n}} C U_Q \Sigma_Q^{+}.$$

It follows that the approximate graph Laplacian is $\tilde{L} = \tilde{U}\tilde{\Sigma}\tilde{U}^T$. Spectral embedding matrix U is computed from the first k eigenvectors of $\tilde{U}$, and its rows are normalized. As in the ordinary spectral clustering algorithm, k-means is applied to the rows of U.

Equipped with the fast spectral clustering algorithm, we will modify the spectral clustering algorithm for the m-layer graph $M_G = \{(G_i = (V, E_i, W_i))_{i\le m}\}$ so that its complexity is linear in n. The first step is to compute the approximate sampled Laplacian C_i for each layer and approximate spectral embedding U_i as in the work of Choromanska *et al.* (2013). The next step is computation of the approximate modified Laplacian (5) based on $\mathcal{L}$. Note that the set of sampled column indices $\mathcal{L}$ is the same through all graph layers:

$$\tilde{C}_{mod} = \sum_{i=1}^{m} C_i + \alpha \sum_{i=1}^{m} U_i U_i^T(:, \mathcal{L}). \tag{15}$$

Since we do not want the complexity of our algorithm to be squared in n, we cannot first compute $U_i U_i^T$ and then sample its columns, but we compute the sampled matrix immediately:

$$U_i U_i^T(:, j) = \sum_{t=1}^{k} U_i(:, t) U_i(\mathcal{L}_j, t). \tag{16}$$

Using (12), we will now compute an upper bound for the error of the approximation $\tilde{L}_{mod}$. As in the work of Choromanska *et al.* (2013), assume that for each layer graph Laplacian has one on its main diagonal and obtain the following bound.

Theorem 2. *Let $\{L_i \in \mathbb{R}^{n\times n}\}_{i=\overline{1,m}}$ be the set of ideal graph Laplacians of each layer. Assume that l columns of L_i are sampled uniformly at random without replacements, and that $\tilde{L}_{mod}$ is the best r- rank Nystrom approximation of L_{mod}. Let*

$$\epsilon > 0, \quad l \ge \frac{64r}{\epsilon^4}, \quad \eta = \sqrt{\frac{\log(\frac{2}{\delta})\xi(l, n-l)}{l}},$$

where $\delta \in (0, 1)$ and

$$\xi(a, b) = \frac{ab}{a+b-1/2} \frac{1}{1 - \frac{1}{2\max(a,b)}}.$$

With probability $1-\delta$ the following expression is true:

$$||L_{mod} - \tilde{L}_{mod}||_F \le \epsilon \Big[\Big(1 + \frac{\alpha m k}{l}\Big)\Big(\eta + \alpha\eta k + m + \frac{\alpha m k}{n}\Big)\Big]. \tag{17}$$

Proof. From Theorem 1 we have

$$||L_{mod} - \tilde{L}_{mod}||_F \le \epsilon \Big[\Big(\frac{n}{l}\gamma\Big)\Big(\theta + \eta \max_{i=\overline{1,n}} n(L_{mod})_{ii}\Big)\Big]^{\frac{1}{2}}, \tag{18}$$

where $\gamma = \sum_{i\in D(l)} (L_{mod})_{ii}$, $D(l)$ is the set of indices of l largest diagonal elements of L_{mod}, and $\theta = \sqrt{n\sum_{i=1}^{n}(L_{mod})_{ii}^2}$.

We will first compute upper bounds to γ, θ and $\max_{i=\overline{1,n}} n(L_{mod})_{ii}$:

$$\begin{aligned}
\gamma &= \sum_{j\in D(l)} (L_{mod})_{jj} \\
&= \sum_{j\in D(l)} \Big(\sum_{i=1}^{m} (L_i)_{jj} + \alpha \sum_{i=1}^{m} (U_i U_i^T)_{jj}\Big) \\
&\le ml + \alpha \sum_{i=1}^{m} \mathrm{tr}(U_i U_i^T) \le ml + \alpha m k,
\end{aligned} \tag{19}$$

$$\begin{aligned}
\theta &= \sqrt{n\sum_{j=1}^{n} (L_{mod})_{jj}^2} \\
&\le \sum_{j=1}^{n} (L_{mod})_{jj} \\
&= \sum_{j=1}^{n} \Big(\sum_{i=1}^{m} (L_i)_{jj} + \alpha \sum_{i=1}^{m} (U_i U_i^T)_{jj}\Big) \\
&= mn + \alpha \sum_{i=1}^{m} \mathrm{tr}(U_i U_i^T) \\
&= mn + \alpha m k,
\end{aligned} \tag{20}$$

$$\begin{aligned}
\max_{j=\overline{1,n}} n(L_{mod})_{jj} &= n \max_{j=\overline{1,n}} \sum_{i=1}^{m} (L_i)_{jj} + \alpha \sum_{i=1}^{m} (U_i U_i^T)_{jj} \\
&\le n + n\alpha k.
\end{aligned} \tag{21}$$

Applying (19), (20) and (21) to (18), we get

$$||L_{mod} - \tilde{L}_{mod}||_F \le n\epsilon \Big[\Big(1 + \frac{\alpha m k}{l}\Big)\Big(\eta + \alpha\eta k + m + \frac{\alpha m k}{n}\Big)\Big]^{\frac{1}{2}}. \tag{22}$$

■

Now, using the same idea regarding the

Algorithm 4. Fast constrained spectral clustering (FAST-CSP-ML).

Input: W, l, k and $\mathcal{PC}$;
Sample column indices $\mathcal{L}$;
Create sampled layers of multi-layer graph;
For each i compute sampled L_i and sampled U_i;
Compute C_{mod} and $Q = U_Q \Sigma_Q U_Q^T$;
Compute $\tilde{\Sigma}$, $\tilde{U}$;
Compute U from $\tilde{U}$, normalize and cluster its rows;
Output: Clusters $C_1, \ldots, C_k$

representation of constraints via a graph layer as in the previous chapter, we implement (4) for fast constrained spectral clustering. Algorithm 4 first converts constraints into the data layers and then performs fast spectral clustering of a multi-layer graph based on a Nyström method. The complexity of this algorithm is $\mathcal{O}(nl^2+t_{km})$, while that of Algorithm 3 is $\mathcal{O}(n^2k+t_{km})$.

Table 1. UCI data sets.

Data set	Instances	Attributes
Iris	100	4
Wine	130	13
Glass	214	9
Seeds	210	7
Ionosphere	351	34
Sylva	145 252	216
Orange	50 000	43
Sdd	58 509	49
Postures	78 095	38
Covtype	581 012	54

5. Experimental results

In this section we evaluate the performance of our algorithms, constrained spectral-multilayer (CSP-ML) (Algorithm 3) and fast-constrained spectral-multilayer (FAST-CSP-ML) (Algorithm 4). We compare these algorithms with the constrained spectral clustering algorithm (CSP) by Wang *et al.* (2014) and the spectral learning algorithm (SL) by Kamvar *et al.* (2003). Matlab code for CSP has been taken from the work of Wang (2014). We compared the algorithms on a machine with an Intel Core i7-4790 CPU 360 GHz, 4 cores and 32 GB memory.

We use 10 real world data sets, and all of them are from the work of Lichman (2013): Wine (Section 5.1), Iris (Section 5.1), Seeds (Sections 5.1–5.5), Glass (Sections 5.1–5.5), Ionosphere (Sections 5.1– 5.5), Sensorless drive diagnosis (Sdd, Section 5.6), Postures (Section 5.6) and Covtype (Section 5.6) except for a Orange[1](only attributes without missing data) data set and the Sylva[1] data set. . We form the adjacency matrix using an RBF kernel,

$$k(x,y) = \exp(-\frac{1}{2}||x-y||^2).$$

Constraints are derived from the exact labels. For example, we choose a subset A of input feature vectors and we assume that we know their labels. If items $v_i, v_j \in A$, then

1. if they belong to the same class, then we add $\{v_i, v_j, t, ML\}$ to $\mathcal{PC}$;
2. if they belong to different classes, then we add $\{v_i, v_j, t, CL\}$ to $\mathcal{PC}$.

For information about data sets, see Table 1. Results were evaluated using the Rand index as defined by Manning *et al.* (2008):

$$RI = \frac{tp+tn}{tp+fp+tn+fn},$$

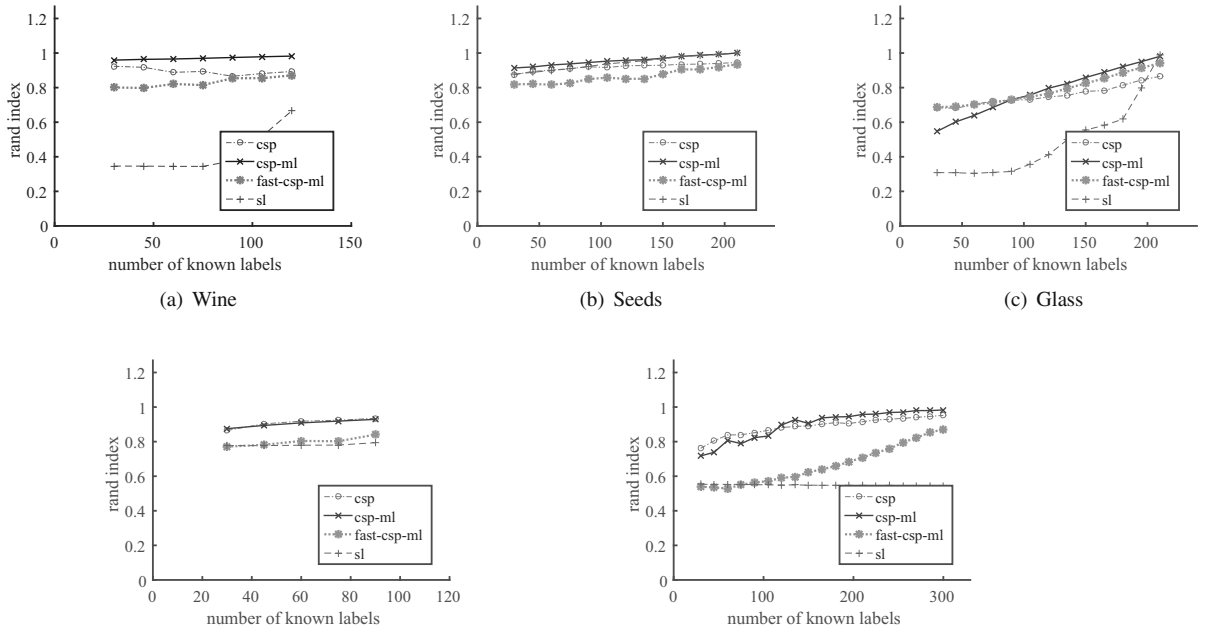
where tp, fp, tn, fn stand for *true positive*, *false positive*, *true negative* and *false negative*, respectively.

Across all figures, the number of the x-coordinate represents known instances and y-axis represents the rand index, except for the soft edge constraints test where the number of the x-coordinate represent the number of known edges.

5.1. Semi supervised setting. We start with 30 instances with known labels and increase the known labels by 15 at each step. At each step we perform 30 tests (in each test we randomly select known labels) and report the mean Rand index. Let us explain the results in Fig. 1:

- Our algorithms CSP-ML and FAST-CSP-ML perform well across all data sets. They effectively utilize a small and a large number of constraints. With a small number of given constraints, the algorithms are able to produce good results, and they consistently give better results as more constraints are added. Furthermore, the cluster partition derived from our algorithm converges to the true partition.
- Our algorithm CSP-ML is able to outperform or produce results comparable to its competitors, the constrained spectral clustering algorithm (Wang *et al.*, 2014) and the spectral learning one (Kamvar *et al.*, 2003), across all data sets. We deduce that the greatest margin between our algorithm CSP-ML and its competitors is achieved when a large number of instances is known. On the other hand, our algorithm FAST-CSP-ML is able to produce better results than constrained spectral clustering only on the Glass data

[1] http://www.causality.inf.ethz.ch/activelearning.php.



(a) Wine (b) Seeds (c) Glass

(d) Iris (e) Ionosphere

Fig. 1. Performance comparison of different constrained spectral clustering algorithms on real world data sets.

set and better than spectral learning only on the Glass, Wine and Ionosphere datasets. However, the greatest advantage of algorithm FAST-CSP-ML is its complexity, $\mathcal{O}(nl^2 + t_{km})$ compared with that of other algorithms, which is squared in the number of instances.

Additionally, we report the average Rand index, its standard deviation and the normalized mutual information, its standard deviation in the Table 2. The normalized mutual information (*nmi*) between random variables X and Y is calculated in the following way:

$$NMI(X,Y) = \frac{I(X;Y)}{\sqrt{H(X)H(Y)}},$$

where $I(X;Y)$ is the mutual information and $H(X)$ is the entropy. Due to the space limitation and the fact that the standard deviation is low and similar in all of the remaining experiments, we present the results via figures only.

5.2. Inconsistent constraints. If there is no cluster assignment that satisfies the entire set of constraints $\mathcal{PC}$, then the set $\mathcal{PC}$ contains inconsistent constraints. For example, inconsistency can appear because of errors in a set of labels. In this paper, we will consider the following type of inconsistent constraints. Let a, b and c be instances from the data set. The following set of constraints is inconsistent: $\{ML(a,b), ML(b,c), CL(a,c)\}$.

Three data sets (Glass, Seeds and Ionosphere) are used for testing in this setting. We start with 30 instances with known labels and increase the known labels by 15 at each step. At each step x we perform 30 tests (in each test we randomly select x known labels and randomly choose $0.08x$ inconsistent constraints) and report the mean rand index. Let us explain the results in Fig. 2:

- Across all three data sets our algorithms CSP-ML and FAST-CSP-ML are able to effectively utilize constraints with inconsistency. These algorithms are able to utilize a small number of constraints and adding constraints improves results. CSP-ML is able to converge to the true cluster partition.

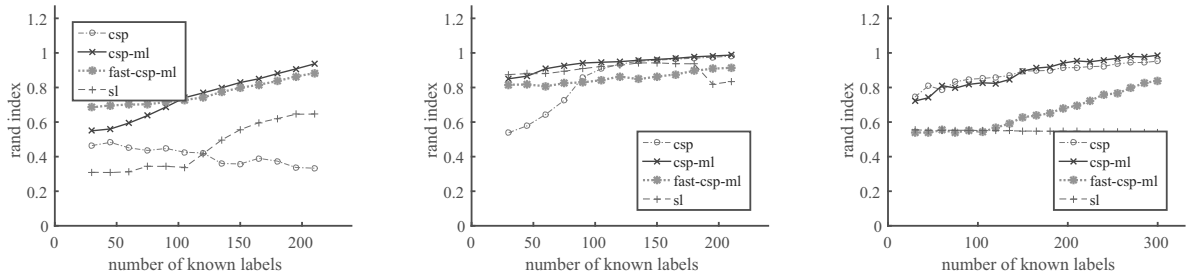
- The algorithm CSP-ML either outperforms competitors or produces slightly weaker results. On the other hand, FAST-CSP-ML produces the best results on the Glass data set, worse results on Seeds and better results than the spectral learning algorithm but worse than constrained spectral clustering on the Ionosphere data set.

5.3. Constrained set with 30% noise. In this case we assume that the set of constraints has 30% noise. Constraint $CL(a,b)$ is noisy if instances a and b belong to the same cluster. We use three data sets (Glass, Seeds and Ionosphere) in this case.

We start with 30 instances with known labels and increase the known labels by 15 at each step. At each step x we perform 30 tests (in each test we randomly select x known labels and from these known labels we induce 70%

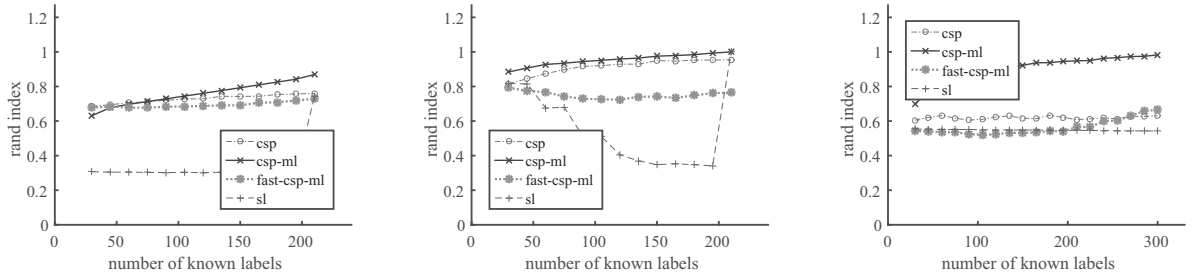
Table 2. Performance comparison of different constrained spectral clustering algorithms on the following real world data sets: Glass, Seeds and Ionosphere. We report an average Rand index and its standard deviation, as well as average normalized mutual information and its standard deviation.

Known labels	Measure of comparison							
	RI				NMI			
Algorithm	CSP	CSP-ML	FAST-CSP-ML	SL	CSP	CSP-ML	FAST-CSP-ML	SL
Data set	Glass							
30	0.68 ± 0.12	0.55 ± 0.13	0.69 ± 0.12	0.31 ± 0.06	0.36 ± 0.08	0.35 ± 0.08	0.28 ± 0.06	0.12 ± 0.02
75	0.72 ± 0.13	0.69 ± 0.13	0.71 ± 0.13	0.31 ± 0.07	0.39 ± 0.10	0.49 ± 0.10	0.33 ± 0.08	0.12 ± 0.02
120	0.75 ± 0.14	0.80 ± 0.14	0.76 ± 0.14	0.41 ± 0.14	0.44 ± 0.11	0.61 ± 0.11	0.46 ± 0.09	0.32 ± 0.20
165	0.78 ± 0.14	0.89 ± 0.16	0.85 ± 0.15	0.58 ± 0.14	0.53 ± 0.11	0.76 ± 0.14	0.62 ± 0.12	0.55 ± 0.14
Data set	Seeds							
30	0.87 ± 0.16	0.91 ± 0.16	0.82 ± 0.15	0.88 ± 0.16	0.71 ± 0.15	0.76 ± 0.14	0.53 ± 0.14	0.71 ± 0.13
75	0.91 ± 0.16	0.94 ± 0.17	0.83 ± 0.16	0.91 ± 0.16	0.75 ± 0.16	0.81 ± 0.15	0.57 ± 0.14	0.78 ± 0.14
120	0.93 ± 0.17	0.96 ± 0.17	0.85 ± 0.17	0.95 ± 0.17	0.81 ± 0.15	0.87 ± 0.16	0.65 ± 0.15	0.85 ± 0.16
165	0.93 ± 0.17	0.98 ± 0.18	0.91 ± 0.17	0.98 ± 0.18	0.83 ± 0.15	0.92 ± 0.17	0.75 ± 0.14	0.92 ± 0.17
Data set	Ionosphere							
30	0.76 ± 0.15	0.72 ± 0.15	0.54 ± 0.10	0.55 ± 0.10	0.46 ± 0.11	0.36 ± 0.18	0.18 ± 0.04	0.09 ± 0.02
75	0.84 ± 0.15	0.79 ± 0.19	0.55 ± 0.11	0.55 ± 0.10	0.58 ± 0.12	0.52 ± 0.22	0.19 ± 0.07	0.08 ± 0.02
120	0.88 ± 0.16	0.90 ± 0.18	0.59 ± 0.12	0.55 ± 0.10	0.64 ± 0.12	0.68 ± 0.18	0.21 ± 0.08	0.07 ± 0.02
165	0.90 ± 0.16	0.94 ± 0.17	0.64 ± 0.14	0.55 ± 0.10	0.71 ± 0.13	0.80 ± 0.15	0.31 ± 0.09	0.06 ± 0.02



(a) Glass (b) Seeds (c) Ionosphere

Fig. 2. Performance comparison of different constrained spectral clustering algorithms on data sets with inconsistent constraints.

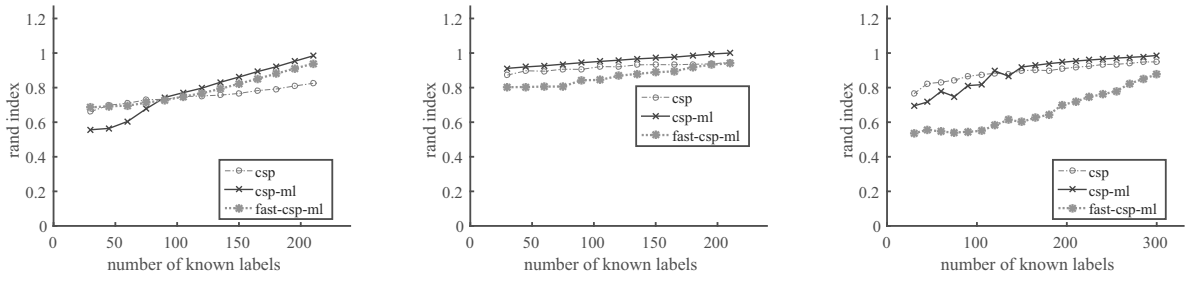


(a) Glass (b) Seeds (c) Ionosphere

Fig. 3. Performance comparison of different constrained spectral clustering algorithms on a constrained data set with noise.

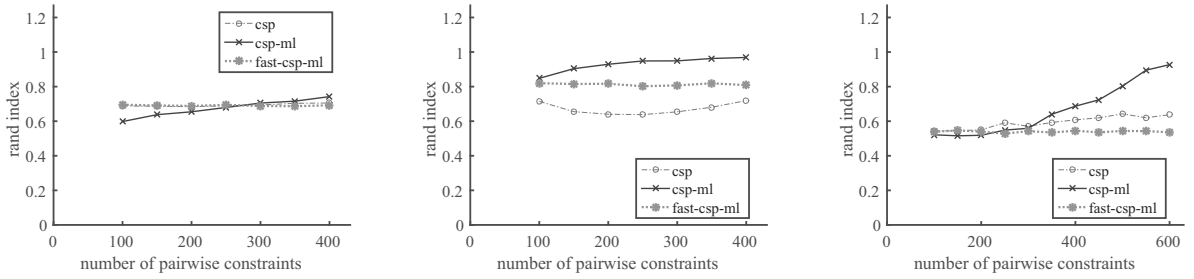
correct constraints and 30% noisy constraints) and report the mean Rand index. Let us look at the results in Fig. 3:

- Across all three data sets our algorithm CSP-ML is able to effectively utilize constraints with 30% noise and converge to the true partition. On the other hand, FAST-CSP-ML usually gives the same results regardless of the number of constraints.
- The algorithm CSP-ML outperforms competitors across all three data sets. We noticed that, on the Ionosphere data set (sparse graph), CSP-ML



(a) Glass (b) Seeds (c) Ionosphere

Fig. 4. Performance comparison of different constrained spectral clustering algorithms on a data set with soft constraints.



(a) Glass (b) Seeds (c) Ionosphere

Fig. 5. Performance comparison of different constrained spectral clustering algorithms on a data set with soft edge constraints.

outperforms competitors and FAST-CSP-ML by a large margin. On the other hand, FAST-CSP-ML produces better results than the spectral learning algorithm but worse than the constrained spectral clustering one across all three data sets.

5.4. Soft constraints. So far we have only performed tests on sets with hard constraints, but here we will do experiments on examples with soft constraints. We use three data sets (Glass, Seeds and Ionosphere) in this case. Since the spectral learning algorithm is unable to utilize soft constraints, we compare our algorithms CSP-ML and FAST-CSP-ML only with the constrained spectral clustering one (Wang *et al.*, 2014).

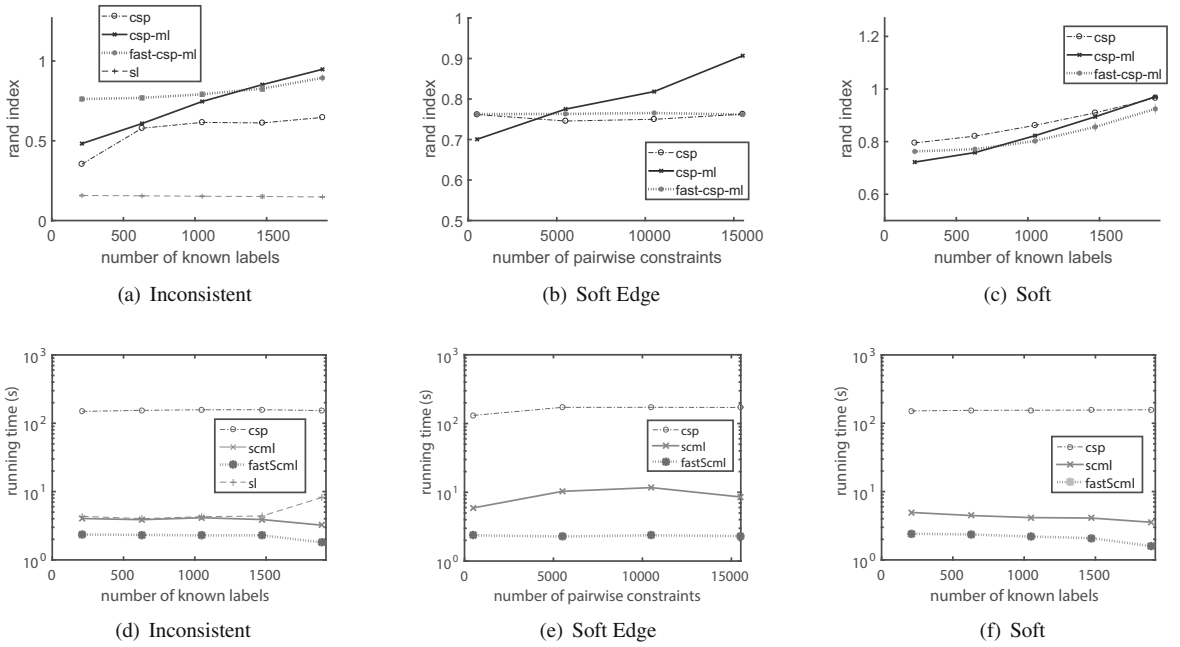
We start with 30 instances with known labels and increase the known labels by 15 at each step. At each step x we perform 30 tests (at each test we randomly select x known labels and from these known labels we induce constraints with a degree between 0.7 and 1.0) and report the mean Rand index. From Fig. 4 we conclude what follows:

- Across all three data sets our algorithms CSP-ML and FAST-CSP-ML are able to effectively utilize soft edge constraints. They are able to utilize a small number of constraints and adding constraints, the produces better results. Additionally, CSP-ML is able to converge to the true cluster partition.
- The algorithm CSP-ML produces slightly better results than its competitors across all data set except for a small number of known labels on the Glass and Ionosphere data sets. On the other hand, FAST-CSP-ML produces weaker results than its competitors on the Seeds and Ionosphere data sets and similar results to its best competitor on the Glass set.

5.5. Soft edge constraints. So far we have tested sets with pairwise constraints induced from a set of known labels. However, now we will randomly choose a set of pairwise constraints from a set of all constraints. Furthermore, all constraints will be soft. Therefore, since the spectral learning algorithm is unable to utilize soft constraints, we compare our algorithms CSP-ML and FAST-CSP-ML only with the constrained spectral clustering algorithm (Wang *et al.*, 2014).

Each constraint will be paired with a randomly chosen degree between 0.7 and 1.0. The X-coordinate will represent the number of known pairwise constraints, and the Y-axis represents the rand index. Let us look at the result form Fig. 5:

- Across all three data sets, our algorithm CSP-ML is



(a) Inconsistent (b) Soft Edge (c) Soft

(d) Inconsistent (e) Soft Edge (f) Soft

Fig. 6. Performance and running time comparison of different constrained spectral clustering algorithms on a subset of covtype data with three different types of constraints (inconsistent, soft and soft edge). Running time is presented with a logarithmic scale.

able to effectively utilize soft edge constraints. It can process a small number of constraints and adding constraints produces better results. Additionally, CSP-ML is able to converge to the true cluster partition.

- The algorithm CSP-ML outperforms its competitors or produces comparably similar results. Specifically, on the Ionosphere data set it outperforms competitors by a large margin. On the other hand, FAST-CSP-ML produces better results than constrained spectral clustering on the Seeds data set and similar results on Glass and Ionosphere data sets.

5.6. Large input set. In this part we test our algorithms on large data sets. First, we empirically show advantages that our algorithms exhibit on the covtype (Lichman, 2013) data set. Covtype is a large data set that contains 581 012 items with 54 attributes that belong to 7 different classes. In our experiments we only used 10 quantitative attributes with largest variance across the data set. This data set will be covtype10.

Because of the memory complexity of $\mathcal{O}(n^2)$ implementation of algorithms CSP-ML, CSP and SL requires at least 2 TB of memory. On the other hand, algorithm FAST-CSP-ML would not have such problems because its memory complexity is $\mathcal{O}(kn)$. Therefore, in order to compare these algorithms, we used a subset of the covtype10 data set with 2100 items (300 belonging to each of the different 7 classes) chosen at random out of 581 012 items in total. Furthermore, we performed tests with inconsistent (Section 5.2), soft (Section 5.4) and soft edge (Section 5.5) constraints and we compared both performance of the algorithms and their running times. Let us look at the results from the Fig. 6:

- On a set with inconsistent constraints, both our algorithms CSP-ML and FAST-CSP-ML produced the best results regardless of the size of the set of constraints.
- On a set with soft constraints, all three algorithms (SL is not applicable to a set with soft constraints) produced similar results, with CSP being slightly better than its competitors.
- On a set with soft edge constraints, FAST-CSP-ML and CSP produced similar results, while CSP-ML produced the worst results on a small set of constraints and the best results on a large set of constraints.
- In every test with all three types of constraints, CSP is a lot slower compared to the other algorithms.
- The benefit of the linear time and memory complexity of the FAST-CSP-ML algorithm becomes obvious while clustering the entire covtype data set.

Next we performed tests on the entire covtype and covtype10 data set as well as other large data sets: sdd, postures, orange and sylva. As already mentioned, only the FAST-CSP-ML algorithm is applicable to data sets as

Table 3. Performance of the FAST-CSP-ML algorithm on large data sets with two different types of constraints (soft without noise and soft with 30% noise). We report an average Rand index and running time in seconds out of 20 performed tests.

Known labels (%)	Noisy constraints No RI	Time (s)	Noisy constraints Yes RI	Time (s)	Noisy constraints No RI	Time (s)	Noisy constraints Yes RI	Time (s)	Noisy constraints No RI	Time (s)	Noisy constraints Yes RI	Time (s)
Data set	Covtype10				Covtype				Sylva			
10%	0.59 ± 0.01	27.93	0.58 ± 0.01	26.27	0.48 ± 0.01	20.23	0.46 ± 0.01	22.32	0.51 ± 0.01	1.13	0.50 ± 0.01	1.12
30%	0.59 ± 0.01	27.03	0.59 ± 0.01	30.85	0.47 ± 0.02	23.42	0.49 ± 0.01	21.11	0.54 ± 0.03	1.11	0.50 ± 0.01	1.26
50%	0.62 ± 0.03	29.39	0.59 ± 0.01	33.11	0.52 ± 0.05	21.34	0.51 ± 0.04	23.19	0.64 ± 0.02	1.16	0.52 ± 0.01	1.29
70%	0.68 ± 0.04	34.03	0.59 ± 0.01	29.49	0.52 ± 0.05	22.11	0.53 ± 0.05	23.09	0.73 ± 0.01	1.18	0.58 ± 0.03	1.39
90%	0.70 ± 0.04	27.90	0.58 ± 0.01	32.65	0.65 ± 0.06	21.51	0.54 ± 0.03	21.78	0.91 ± 0.01	1.18	0.67 ± 0.04	1.52
Data set	Orange				Sdd				Postures			
10%	0.50 ± 0.01	1.06	0.50 ± 0.01	0.89	0.83 ± 0.01	2.41	0.84 ± 0.01	2.15	0.70 ± 0.01	2.33	0.68 ± 0.01	2.15
30%	0.54 ± 0.01	0.97	0.51 ± 0.01	1.02	0.84 ± 0.01	2.39	0.84 ± 0.01	2.41	0.70 ± 0.01	2.36	0.69 ± 0.01	2.20
50%	0.62 ± 0.01	0.97	0.56 ± 0.02	1.31	0.87 ± 0.01	2.23	0.84 ± 0.01	3.74	0.74 ± 0.02	2.19	0.69 ± 0.01	2.85
70%	0.74 ± 0.01	0.93	0.66 ± 0.02	1.54	0.90 ± 0.01	2.20	0.84 ± 0.01	5.75	0.80 ± 0.03	2.05	0.70 ± 0.01	3.49
90%	0.89 ± 0.01	0.91	0.74 ± 0.02	2.55	0.96 ± 0.01	1.86	0.85 ± 0.01	7.43	0.87 ± 0.03	1.79	0.72 ± 0.01	4.47

large as these. Li *et al.* (2015) developed an approximate version of the CSP algorithm based on a sparse coding. However, their algorithm requires constrained matrix of size $\mathcal{O}(n^2)$, which is intractable when the number of constraints is large. On the other hand, both time and memory complexities of our algorithm are linear in the size of the data set regardless of the number of constraints, which can be of the order of n^2.

We performed tests on data sets with soft constraints without noise and with 30% noise, and on a set with soft edge constraints. The FAST-CSP-ML algorithm is applied and results are shown in Table 3. We note that the algorithm is able to effectively use constraints, and as we add constraints the results are improved.

6. Conclusion

We proposed a new constrained spectral clustering algorithm based on unconstrained spectral clustering on a multi-layer graph. The algorithm uses graph representation of both data and constraints as different layers of a multi-layer graph, to which unconstrained spectral clustering based on the results of Dong *et al.* (2014) is applied. We validated our algorithm on real world data sets, with different types of constraints (hard, soft, inconsistent and noisy), and we showed that it achieves superior or comparative results to the state-of-the-art algorithm (Wang *et al.*, 2014).

Further, we considered the problem of time complexity of our algorithm, which is squared in the number of items. As a solution to this problem we suggested the use of the Nyström method, which achieves linear time in a number of items. To the best of our knowledge, this is the only constrained spectral clustering algorithm with such complexity that can be applied to both hard and soft constraints regardless of the size of the set of constraints.

Acknowledgment

This research was supported by the Ministry of Education, Science and Technological Development, Republic of Serbia (grant no. 174013).

References

Choromanska, A., Jebara, T., Kim, H., Mohan, M. and Monteleoni, C. (2013). Fast spectral clustering via the Nyström method, *International Conference on Algorithmic Learning Theory, Singapore, Republic of Singapore*, pp. 367–381.

Coleman, T., Saunderson, J. and Wirth, A. (2008). Spectral clustering with inconsistent advice, *25th International Conference on Machine Learning, Helsinki, Finland*, pp. 152–159.

De Bie, T., Suykens, J. and De Moor, B. (2004). Learning from general label constraints, *Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (SSPR), Lisbon, Portugal*, pp. 671–679.

Dong, X., Frossard, P., Vandergheynst, P. and Nefedov, N. (2014). Clustering on multi-layer graphs via subspace analysis on Grassmann manifolds, *IEEE Transactions on Signal Processing* **62**(4): 905–918.

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

Golub, G.H. and Van Loan, C.F. (1996). *Matrix Computations*, Johns Hopkins University Press, Baltimore, MD, pp. 374–426.

Hamm, J. and Lee, D.D. (2008). Grassmann discriminant analysis: A unifying view on subspace-based learning, *Proceedings of the 25th International Conference on Machine Learning, Helsinki, Finland*, pp. 376–383.

Hamm, J. and Lee, D.D. (2009). Extended Grassmann kernels for subspace-based learning, *in* D. Koller *et al.* (Eds.), *Advances in Neural Information Processing Systems 21*, Curran Associates, Inc., Vancouver, pp. 601–608.

Harandi, M.T., Sanderson, C., Shirazi, S. and Lovell, B.C. (2011). Graph embedding discriminant analysis on Grassmannian manifolds for improved image set matching, *IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Colorado Springs, CO, USA*, pp. 2705–2712.

Kamvar, S.D., Klein, D. and Manning, C.D. (2003). Spectral learning, *18th International Joint Conference on Artificial Intelligence, Acapulco, Mexico*, pp. 561–566.

Kumar, S., Mohri, M. and Talwalkar, A. (2009). Sampling techniques for the Nyström method, *12th International Conference on Artificial Intelligence and Statistics, Barcelona, Spain*, pp. 304–311.

Li, J., Xia, Y., Shan, Z. and Liu, Y. (2015). Scalable constrained spectral clustering, *IEEE Transactions on Knowledge and Data Engineering* **27**(2): 589–593.

Li, M., Lian, X.-C., Kwok, J.T. and Lu, B.-L. (2011). Time and space efficient spectral clustering via column sampling, *IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Colorado Springs, CO, USA,* pp. 2297–2304.

Li, Z., Liu, J. and Tang, X. (2009). Constrained clustering via spectral regularization, *IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, USA*, pp. 421–428.

Lichman, M. (2013). UCI machine learning repository, University of California Irvine, Irvine, CA, `http://archive.ics.uci.edu/ml`.

Lu, Z. and Carreira-Perpinan, M.A. (2008). Constrained spectral clustering through affinity propagation, *IEEE Conference on Computer Vision and Pattern Recognition, Anchorage, AK, USA*, pp. 1–8.

Manning, C.D., Raghavan, P. and Schütze, H. (2008). *Introduction to Information Retrieval*, Cambridge University Press, New York, NY.

Ng, A.Y., Jordan, M.I. and Weiss, Y. (2002). On spectral clustering: Analysis and an algorithm, *Advances in Neural Information Processing Systems* **2**(14): 849–856.

Turk, M. and Pentland, A. (1991). Eigenfaces for recognition, *Journal of Cognitive Neuroscience* **3**(1): 71–86.

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

Wang, X. (2014). On constrained spectral clustering and its applications (Matlab code), `https://sites.google.com/site/gnaixgnaw/home`.

Wang, X., Qian, B. and Davidson, I. (2014). On constrained spectral clustering and its applications, *Data Mining and Knowledge Discovery* **28**(1): 1–30.

White, S. and Smyth, P. (2005). A spectral clustering approach to finding communities in graphs, *Proceedings of the 2005 SIAM International Conference on Data Mining, Newport Beach, CA, USA*, pp. 274–285.

Xu, Q., des Jardins, M. and Wagstaff, K. (2005). Constrained spectral clustering under a local proximity structure assumption, *18th International Conference of the Florida Artificial Intelligence Research Society (FLAIRS-05), Clearwater Beach, FL, USA*, pp. 866–867.

Aleksandar Trokicić is currently a teaching assistant and a PhD student at the Faculty of Mathematics and Sciences, University of Niš, Serbia. He holds an MSc degree in computer science from the same university. His research interests include machine learning, semi-supervised learning, clustering, randomized learning, fast spectral learning and its applications.

Branimir Todorović is an associate professor at the Computer Science Department, Faculty of Mathematics and Sciences, University of Niš. He holds a DSc degree from the Faculty of Electrical Engineering, University of Belgrade. His research interest include sequential Bayesian training of feed forward and recurrent neural networks, blind source separation and deconvolution, on-line training of structural classifiers, active and semi-supervised learning algorithms.

Received: 2 February 2018
Revised: 1 September 2018
Accepted: 16 October 2018