Hypergraph 1-Spectral Clustering with General Submodular Weights

Yu Zhu, Boning Li, and Santiago Segarra
Dept. of Electrical and Computer Engineering, Rice University, USA

***Abstract*—We develop a spectral clustering algorithm for general submodular hypergraphs based on their 1-Laplacians. More precisely, we utilize the eigenvector associated with the second smallest eigenvalue of the hypergraph 1-Laplacian to cluster the vertices. The computation of the eigenvector is based on the inverse power method, the key of which is to solve its inner-loop optimization problem. Efficient solutions to this inner problem exist when submodular hypergraphs are equipped with hyperedge splitting functions with special structures such as when they are cardinality based or graph reducible. In this paper, we present a solution to the inner problem for general submodular splitting functions by adopting a random coordinate descent method together with the Fujishige-Wolfe algorithm. Numerical experiments using real-world data demonstrate the effectiveness of the proposed clustering algorithm.**

***Index Terms*—submodular hypergraphs, spectral clustering, 1-Laplacian, hyperedge splitting functions, submodular functions**

I. Introduction

Spectral clustering leverages the eigenvalues and eigenvectors of the graph Laplacian to group the vertices of the graph. It is one of the most popular clustering methods due to its generality, efficiency, and strong theoretical basis. The standard graph Laplacian (2-Laplacian) was first adopted in the context of spectral clustering to obtain relaxations of balanced graph cut criteria [1]. Then, it was generalized to the p-Laplacian, which is able to provide a better approximation of the Cheeger constant [2, 3]. Especially, the second smallest eigenvalue of the 1-Laplacian is equal to the Cheeger constant, and the partition that achieves the optimal normalized Cheeger cut can be obtained via thresholding the corresponding eigenvector [4].

In *graphs*, vertices are assumed to interact via pairwise relations. However, this can be a limitation in a wide range of machine learning tasks where entities may naturally engage in higher-order relations [5, 6]. Therefore, *hypergraphs* emerge as an extension to graphs with the generalized notion of an edge, namely a hyperedge, that can simultaneously connect multiple (≥ 2) vertices. Real-world examples of a hyperedge include a product for which multiple customers place orders [7], a keyword contained in multiple news articles [8, 9], and a publication on which multiple authors collaborate [10].

An edge in a graph connects two vertices, hence there is only one way to split it and a scalar weight is enough to describe the splitting cost. Different from graphs, there may exist different ways to split a hyperedge. In order to assign costs to every possible cut of a hyperedge e, a splitting function w_e is introduced where $w_e(\mathcal{S})$ indicates the penalty of partitioning e into a subset $\mathcal{S} \subseteq e$ and its complement $e \setminus \mathcal{S}$ [11]. When w_e is submodular for every hyperedge e, the corresponding model is called *submodular hypergraph* [12]. Fundamental theoretical results in spectral graph theory including p-Laplacians, nodal domain theorems, and Cheeger inequalities have been generalized to submodular hypergraphs.

The choice of the splitting function has a significant impact on the clustering results. There are three subclasses of submodular splitting functions considered in existing work. One is the so-called all-or-nothing splitting function, which simply resembles the edge weight in graphs and assigns an identical cost if the hyperedge is split regardless of how its vertices are separated [4]. This splitting function disregards the fact that different subsets of vertices belonging to the same hyperedge may have different degrees of importance when clustered together. Consequently, cardinality-based splitting functions are proposed, where the splitting penalty depends only on the number of vertices placed on each side of the split [11]. The limitation of this choice is that all vertices within the same hyperedge are viewed as equal contributors, thus losing the likely critical information regarding the different levels of vertex-to-hyperedge contributions. In this regard, edge-dependent vertex weights (EDVWs) are introduced, where a positive weight $\gamma_e(v)$ is assigned for every hyperedge e and every vertex v in this hyperedge to quantify the contribution of v to e [10], for example, the amount of a product that was purchased by a given customer. EDVWs-based splitting functions in the form of $w_e(\mathcal{S}) = g_e(\sum_{v\in\mathcal{S}} \gamma_e(v))$ are proposed where g_e is a concave function to guarantee that w_e is submodular [13–15].

Since the 1-Laplacian provides the tightest approximation of the Cheeger constant, we are interested in applying 1-spectral clustering to submodular hypergraphs. To this end, we need to compute the eigenvector of the hypergraph 1-Laplacian associated with the second smallest eigenvalue. Computation methods have been developed based on the inverse power method (IPM) [16]. The key step in IPM is solving an inner-loop optimization problem in it. Efficient solutions to this inner problem have been proposed for submodular hypergraphs equipped with cardinality-based [12] and EDVWs-based splitting functions [14]. In this paper, we consider general submodular splitting functions with no special structures. We develop an algorithm to solve the inner problem which combines a random coordinate descent method (RCDM) [17] and the Fujishige-Wolfe algorithm [18–20]. The effectiveness

This work was supported by NSF under award CCF-2008555. E-mails: {yz126, boning.li, segarra}@rice.edu

2022 56th Asilomar Conference on Signals, Systems, and Computers | 978-1-6654-5906-8/22/$31.00 ©2022 IEEE | DOI: 10.1109/IEEECONF56349.2022.10052065

of the proposed algorithm is validated via numeral experiments on real-world datasets.

II. Preliminaries

We briefly review the submodular function and its related concepts in Section II-A and then introduce the submodular hypergraph model and its corresponding 1-Laplacian in Section II-B. Throughout the paper, we assume that the hypergraph is connected.

A. Mathematical preliminaries

For a finite set $\mathcal{V}$, a set function $F : 2^{\mathcal{V}} \to \mathbb{R}$ is called submodular if $F(\mathcal{S}_1 \cup \{u\}) - F(\mathcal{S}_1) \geq F(\mathcal{S}_2 \cup \{u\}) - F(\mathcal{S}_2)$ for every $\mathcal{S}_1 \subseteq \mathcal{S}_2 \subset \mathcal{V}$ and every $u \in \mathcal{V} \backslash \mathcal{S}_2$. Considering a set function $F : 2^{\mathcal{V}} \to \mathbb{R}$ such that $F(\emptyset) = 0$ where $\mathcal{V} = [N] = \{1, 2, \cdots, N\}$, its Lovász extension $f : \mathbb{R}^N \to \mathbb{R}$ is defined as follows. For any $\mathbf{x} \in \mathbb{R}^N$, sort its entries in non-increasing order $x_{i_1} \geq x_{i_2} \geq \cdots \geq x_{i_N}$, where $(i_1, i_2, \cdots, i_N)$ is a permutation of $(1, 2, \cdots, N)$, and set

$$f(\mathbf{x}) = \textstyle\sum_{j=1}^{N-1} F(\mathcal{S}_j)(x_{i_j} - x_{i_{j+1}}) + F(\mathcal{V})x_{i_N}, \quad (1)$$

where $\mathcal{S}_j = \{i_1, \cdots, i_j\}$ for $1 \leq j < N$. A set function F is submodular if and only if its Lovász extension f is convex [21]. For a submodular function F, its base polytope is a convex set defined as

$$\mathcal{B} = \{\mathbf{y} \in \mathbb{R}^N | \mathbf{y}(\mathcal{S}) \leq F(\mathcal{S}) \text{ for all } \mathcal{S} \subseteq \mathcal{V} \text{ and } \mathbf{y}(\mathcal{V}) = F(\mathcal{V})\}, \quad (2)$$

where $\mathbf{y}(\mathcal{S}) = \sum_{i \in \mathcal{S}} y_i$. The Lovász extension and the base polytope are related as $f(\mathbf{x}) = \max_{\mathbf{y} \in \mathcal{B}} \langle \mathbf{y}, \mathbf{x} \rangle$ [21].

B. Submodular hypergraphs

Let $\mathcal{H} = (\mathcal{V}, \mathcal{E}, \mu, \{w_e\})$ denote a submodular hypergraph [12] where $\mathcal{V} = [N]$ is the vertex set and $\mathcal{E}$ represents the set of hyperedges. The function $\mu : \mathcal{V} \to \mathbb{R}_+$ assigns positive weights to every vertex. Each hyperedge $e \in \mathcal{E}$ is associated with a submodular function $w_e : 2^e \to \mathbb{R}_{\geq 0}$ that assigns non-negative costs to each possible partition of the hyperedge e. Moreover, w_e is required to satisfy $w_e(\emptyset) = 0$ and be symmetric so that $w_e(\mathcal{S}) = w_e(e \setminus \mathcal{S})$ for any $\mathcal{S} \subseteq e$. The domain of the function w_e can be extended from 2^e to $2^{\mathcal{V}}$ by setting $w_e(\mathcal{S}) = w_e(\mathcal{S} \cap e)$ for any $\mathcal{S} \subseteq \mathcal{V}$, guaranteeing that the submodularity is maintained.

A cut is a partition of the vertex set $\mathcal{V}$ into two disjoint, non-empty subsets denoted by $\mathcal{S}$ and its complement $\bar{\mathcal{S}} = \mathcal{V} \setminus \mathcal{S}$. The cut weight is defined as the sum of cut costs associated with each hyperedge [12], i.e., $\mathrm{cut}(\mathcal{S}, \bar{\mathcal{S}}) = \sum_{e \in \mathcal{E}} w_e(\mathcal{S})$. The normalized Cheeger cut (NCC) is defined as

$$\mathrm{NCC}(\mathcal{S}) = \frac{\mathrm{cut}(\mathcal{S}, \bar{\mathcal{S}})}{\min\{\mathrm{vol}(\mathcal{S}), \mathrm{vol}(\bar{\mathcal{S}})\}}, \quad (3)$$

where $\mathrm{vol}(\mathcal{S}) = \sum_{v \in \mathcal{S}} \mu(v)$ denotes the volume of $\mathcal{S}$. The Cheeger constant [12, 22, 23] is defined as

$$h_2 = \min_{\emptyset \subset \mathcal{S} \subset \mathcal{V}} \mathrm{NCC}(\mathcal{S}). \quad (4)$$

The solution to (4) provides an optimal partitioning in the sense that we obtain two balanced clusters (in terms of their volume) that are only loosely connected, as captured by a small cut weight. Although many alternative clustering formulations exist [1, 3, 24], in this paper we adopt the minimization of (4) as our objective.

Optimally solving (4) has been shown to be NP-hard for graphs [3, 4, 25], let alone weighted hypergraphs. Hence, different relaxations have been proposed, a popular one being spectral clustering, a relaxation based on the second eigenvector of the graph (or hypergraph) Laplacian [1, 3, 8]. This approach is also theoretically justified through the Cheeger inequality [10, 12, 22, 23], where h_2 is *upper bounded* by a function of the second smallest eigenvalue of the Laplacian. However, it has been proved that the Cheeger constant h_2 is *equal* to the second smallest eigenvalue λ_2 of the hypergraph 1-Laplacian, an alternative (non-linear) operator that generalizes the classical Laplacian (Theorem 4.1 in [12]). Moreover, the corresponding partitioning can be obtained by thresholding the eigenvector associated with λ_2 (Theorem 4.3 in [12]). The 1-Laplacian $\triangle_1$ of a submodular hypergraph is defined as an operator that, for all $\mathbf{x} \in \mathbb{R}^N$, induces

$$\langle \mathbf{x}, \triangle_1(\mathbf{x}) \rangle = \textstyle\sum_{e \in \mathcal{E}} f_e(\mathbf{x}) \equiv Q_1(\mathbf{x}), \quad (5)$$

where f_e is the Lovász extension of w_e. Notice that $\triangle_1$ can be alternatively defined in terms of the subdifferential of f_e [12], but the inner product definition in (5) is more instrumental to our development.

III. 1-Spectral Clustering

A. IPM-based 1-spectral clustering

We study spectral clustering for general submodular hypergraphs leveraging the 1-Laplacian. As mentioned in Section II-B, for submodular hypergraphs, the Cheeger constant h_2 is equal to the second smallest eigenvalue λ_2 of the 1-Laplacian $\triangle_1$. The corresponding optimal bipartition can be obtained by thresholding the eigenvector of $\triangle_1$ associated with λ_2 [12]. This eigenvector can be computed by minimizing

$$R_1(\mathbf{x}) = \frac{Q_1(\mathbf{x})}{\min_{c \in \mathbb{R}} \|\mathbf{x} - c\mathbf{1}\|_{1,\mu}}, \quad (6)$$

where $\|\mathbf{x}\|_{1,\mu} = \sum_{v \in \mathcal{V}} \mu(v)|x_v|$. Given the eigenvector $\mathbf{x}$, a partitioning can be defined as $\mathcal{S} = \{v \in \mathcal{V} \,|\, x_v > t\}$ and its complement, where t is a threshold value. The optimal t can be determined as the one that minimizes the NCC in (3).

The minimization of $R_1(\mathbf{x})$ can be solved based on IPM [12, 16], as outlined in Algorithm 1. Three functions are introduced: $\mu_+(\mathbf{x}) = \sum_{v \in \mathcal{V}: x_v > 0} \mu(v)$, $\mu_-(\mathbf{x}) = \sum_{v \in \mathcal{V}: x_v < 0} \mu(v)$ and $\mu_0(\mathbf{x}) = \sum_{v \in \mathcal{V}: x_v = 0} \mu(v)$. Although this algorithm cannot guarantee convergence to the second eigenvector, the objective $R_1(\mathbf{x})$ is guaranteed to decrease and converge in the iterative process. The algorithm was first proposed for the undirected graph setting [16], then generalized to submodular hypergraphs with cardinality-based splitting functions [12] and splitting functions that are graph reducible, including EDVWs-based splitting functions [14]. The major difference between

Algorithm 1 IPM-based minimization of $R_1(\mathbf{x})$ [12, 16]

1: **Input:** submodular hypergraph $\mathcal{H} = (\mathcal{V}, \mathcal{E}, \mu, \{w_e\})$ with N vertices, accuracy ϵ
2: **Initialization:** non-constant $\mathbf{x} \in \mathbb{R}^N$ subject to $0 \in \operatorname{argmin}_c \|\mathbf{x} - c\mathbf{1}\|_{1,\mu}$, $\lambda \leftarrow R_1(\mathbf{x})$
3: **repeat**
4: for all $v \in \mathcal{V}$, $g_v \leftarrow \begin{cases} \operatorname{sign}(x_v) \cdot \mu(v), & \text{if } x_v \neq 0 \\ \frac{\mu_-(\mathbf{x}) - \mu_+(\mathbf{x})}{\mu_0(\mathbf{x})} \cdot \mu(v), & \text{if } x_v = 0 \end{cases}$
5: $\mathbf{x} \leftarrow \operatorname{argmin}_{\|\mathbf{x}\|_2 \leq 1} Q_1(\mathbf{x}) - \lambda \langle \mathbf{x}, \boldsymbol{g} \rangle$ **(inner problem)**
6: $\mathbf{x} \leftarrow \mathbf{x} - \operatorname{argmin}_c \|\mathbf{x} - c\mathbf{1}\|_{1,\mu}$
7: $\lambda' \leftarrow \lambda, \lambda \leftarrow R_1(\mathbf{x})$
8: **until** $\frac{|\lambda - \lambda'|}{\lambda'} < \epsilon$
9: **Output:** $\mathbf{x}$

these settings lies in how the inner-loop optimization problem (line 5 in Algorithm 1) is solved.

In [12], the authors solved the inner problem using RCDM [17] together with a divide-and-conquer algorithm proposed in [26], and the computational complexity of the latter algorithm depends on the time of solving the problem $\min_{\mathcal{S} \subseteq e} F(\mathcal{S}) \triangleq w_e(\mathcal{S}) + \mathbf{c}(\mathcal{S})$ for an arbitrary vector $\mathbf{c} \in \mathbb{R}^{|e|}$. For a cardinality-based splitting function, the solution to this problem can be found efficiently via a line search even when $|e|$ is large, in which we start from an empty set $\mathcal{S}_0 = \emptyset$, for each step we add one vertex corresponding to the smallest entry in vector $\mathbf{c}$ that has not been added yet in former steps so that we obtain a series of sets $\mathcal{S}_1, \cdots, \mathcal{S}_{|e|}$ where the last one is $\mathcal{S}_{|e|} = e$, finally we compare their objective values $F(\mathcal{S}_i)$ and identify the solution $\mathcal{S}_{i^*}$ leading to the minimum objective value. However, this is not the case for EDVW-based splitting functions or more general ones.

For submodular hypergraphs that are reducible to some (possibly) directed graph defined on a (possibly) augmented vertex set, the paper [14] derives the dual of the inner problem where the dual problem is defined on the graph obtained via the hypergraph-to-graph reduction. Compared with the primal problem, which is convex but non-smooth, the dual problem is convex as well as smooth, and thus it can be efficiently solved using algorithms including FISTA [27, 28] and PDHG [29, 30] with quadratic convergence. Since it is still an open problem if all submodular splitting functions are graph reducible, it is not clear whether this approach works for all submodular hypergraphs or not.

B. Solving the inner problem for general submodular weights

We propose a solution to the inner problem for hypergraphs with general submodular splitting functions with no special structures. According to Theorem 4.7 in [12], the inner problem is the dual of the following problem

$$\min_{\{\mathbf{y}_e\}} \| \sum_{e \in \mathcal{E}} \mathbf{y}_e - \lambda \mathbf{g} \|_2^2, \quad \mathbf{y}_e \in \mathcal{B}_e, \text{ for all } e \in \mathcal{E}, \tag{7}$$

Algorithm 2 Fujishige-Wolfe algorithm for solving $\min_{\mathbf{y} \in \mathcal{B}} \|\mathbf{y}\|_2$

1: **Input:** base polytope $\mathcal{B}$ of some submodular function F, accuracy ϵ
2: **Initialization:** take any $\mathbf{q} \in \mathcal{B}$, $\mathbf{y} \leftarrow \mathbf{q}$, $\mathcal{S} \leftarrow \{\mathbf{q}_1 = \mathbf{q}\}$, $a_1 \leftarrow 1$; always maintain $\mathbf{y} = \sum_{\mathbf{q}_i \in \mathcal{S}} a_i \mathbf{q}_i$
3: **while** true (major cycle)
4: (a) $\mathbf{q} \leftarrow \operatorname{argmin}_{\mathbf{q} \in \mathcal{B}} \mathbf{y}^\top \mathbf{q}$
5: (b) **if** $\|\mathbf{y}\|_2^2 \leq \mathbf{y}^\top \mathbf{q} + \epsilon^2$ **then** break
6: (c) $\mathcal{S} \leftarrow \mathcal{S} \cup \{\mathbf{q}\}$
7: (d) **while** true (minor cycle)
8: $\mathbf{z}, \text{coefficients}\{b_1, b_2, \cdots\} \leftarrow \operatorname{argmin}_{\mathbf{z} \in \text{aff}(\mathcal{S})} \|\mathbf{z}\|_2$
9: **if** all coefficients $b_i \geq 0$ **then** break
10: $\theta \leftarrow \min_{i: b_i < 0} \frac{a_i}{a_i - b_i}$
11: $\mathbf{y} \leftarrow \theta \mathbf{z} + (1 - \theta)\mathbf{y}$
12: $a_i \leftarrow \theta b_i + (1 - \theta) a_i$
13: $\mathcal{S} \leftarrow \{\mathbf{q}_i : a_i > 0\}$
14: (e) $\mathbf{y} \leftarrow \mathbf{z}$
15: **Output:** $\mathbf{y}$

Algorithm 3 Greedy method for solving $\min_{\mathbf{q} \in \mathcal{B}} \mathbf{y}^\top \mathbf{q}$

1: **Input:** vector $\mathbf{y}$ (assuming its length to be N), $\mathcal{B}$ is the base polytope of a submodular function F
2: sort the entries in $\mathbf{y}$: $y_{i_1} \leq y_{i_2} \leq \cdots \leq y_{i_N}$
3: set $\mathcal{S}_0 = \emptyset$ and $\mathcal{S}_j = \{i_1, \cdots, i_j\}$ for $j = 1, \cdots, N$
4: set the jth entry of $\mathbf{q}$ as $q_j = F(\mathcal{S}_j) - F(\mathcal{S}_{j-1})$
5: **Output:** $\mathbf{q}$

where $\mathcal{B}_e$ is the base polytope of w_e. The primal and dual variables are related as $\mathbf{x} = \frac{\lambda \mathbf{g} - \sum_{e \in \mathcal{E}} \mathbf{y}_e}{\|\lambda \mathbf{g} - \sum_{e \in \mathcal{E}} \mathbf{y}_e\|_2}$.

By leveraging RCDM [17], we sample a hyperedge to optimize in each step. Then in each step, we need to solve a decomposed problem in the following form,

$$\min_{\mathbf{y}_e \in \mathcal{B}_e} \|\mathbf{y}_e + \mathbf{c}\|_2^2, \quad \text{where } \mathbf{c} = \sum_{e' \in \mathcal{E} \setminus e} \mathbf{y}_{e'} - \lambda \mathbf{g}. \tag{8}$$

Set $\mathbf{y}'_e = \mathbf{y}_e + \mathbf{c}$ and rewrite the above problem as

$$\min_{\mathbf{y}'_e \in \mathcal{B}'_e} \|\mathbf{y}'_e\|_2^2, \tag{9}$$

where $\mathcal{B}'_e$ is the base polytope of the submodular function $w_e(\mathcal{S}) + \mathbf{c}(\mathcal{S})$. This problem can be solved using the Fujishige-Wolfe algorithm [19, 20] as summarized in Algorithm 2. For the problem in line 4, it can be solved using the greedy method presented in Algorithm 3. For the problem in line 8, the affine hull of a finite set $\mathcal{S} \subseteq \mathbb{R}^N$ is $\text{aff}(\mathcal{S}) = \{\mathbf{z} | \mathbf{z} = \sum_{\mathbf{q}_i \in \mathcal{S}} b_i \mathbf{q}_i \text{ and the sum of all } b_i \text{ equals } 1\}$. Define the matrix $\mathbf{Q}$ which collects all $\mathbf{q}_i \in \mathcal{S}$ into its columns. Then the vector $\mathbf{b}$ collecting all coefficients b_i can be naively computed as $\mathbf{b} = \frac{(\mathbf{Q}^\top \mathbf{Q})^{-1} \mathbf{1}}{\mathbf{1}^\top (\mathbf{Q}^\top \mathbf{Q})^{-1} \mathbf{1}}$ and $\mathbf{z} = \mathbf{Q}\mathbf{b}$.

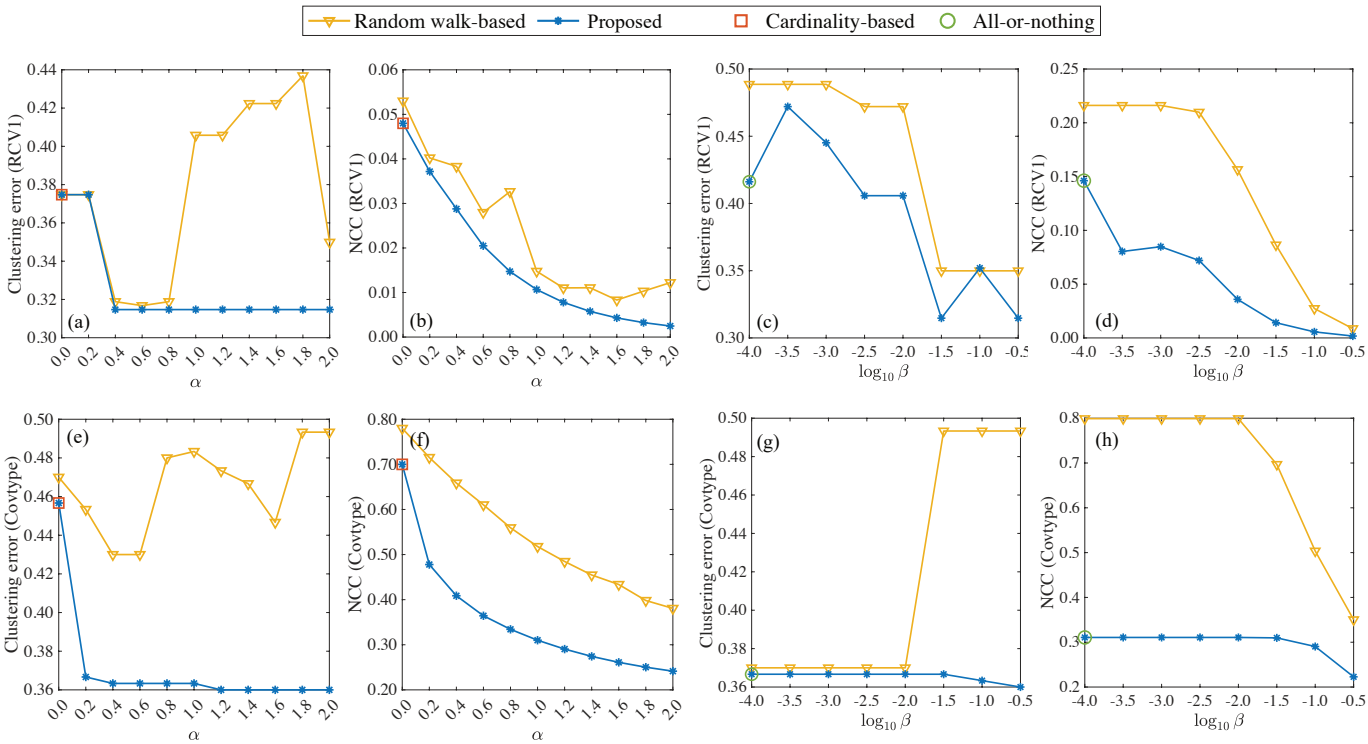

Fig. 1: Clustering performance in two real-world datasets as a function of parameters α and β. On RCV1 dataset, (a-b) plot the clustering error and NCC versus α; (c-d) plot the clustering error and NCC versus $\log_{10}\beta$. Similar results on Covtype dataset are shown in (e-h).

IV. Numerical Experiments

In evaluating the proposed method for hypergraphs with general submodular weights, we focus on the 2-partition case[1]. We consider hyperedge splitting functions in the form of

$$w_e(\mathcal{S}) = \min\{\sum_{v\in\mathcal{S}}\gamma_e(v)\cdot\sum_{v\in e\setminus\mathcal{S}}\gamma_e(v),\ \beta\cdot\sum_{v\in e}\gamma_e(v)\}. \quad (10)$$

Datasets. We consider two widely used real-world datasets.

Reuters Corpus Volume 1 (RCV1)[2]: A collection of manually categorized newswire stories [31]. We consider the documents in categories C22 (new products/services) and C23 (research/development) for our 2-partition evaluation. In modeling the hypergraph, each document is represented by a unique vertex, which belongs to one or more hyperedge(s) manifesting individual words in that document. We select the 30 most frequent words in the corpus, excluding stop words (e.g., articles, pronouns, etc.), common words (found in more than 2% documents), and rare words (found in fewer than 0.1% documents). We also disregard documents containing less than 5 selected words, leaving us with 483 documents and 30 hyperedges (words). The EDVWs $\gamma_e(v)$ are defined as the corresponding term frequency-inverse document frequencies (tf-idf) [32] to the power of an adjustable parameter α.

Covertype Data Set (Covtype)[3]: Areas of different forest cover types with each numerical feature quantized into 15 evenly sized bins and then mapped to hyperedges. We consider two cover type classes, namely the types 4 (Cottonwood/Willow) and 5 (Aspen). We randomly select a subset of 150 vertices from each class, resulting in a downsampled hypergraph of 300 vertices and 144 hyperedges. Its EDVWs are computed as $\exp(-\alpha d_e)$, where d_e is the distance between each feature value in hyperedge/bin e and the median of the bin, normalized to the range $[0, 1]$. Hence, 'typical' vertices (i.e., whose feature values are close to the typical feature value in the corresponding hyperedge/bin) are associated with higher EDVWs.

Baselines. We illustrate the clustering performance of our proposed method by comparing it with three baseline approaches.

Random walk-based: A hypergraph Laplacian is proposed by [8] based on random walks with EDVWs. We adopt it as a competitor method and, to get the partitioning, threshold the second eigenvector of the normalized hypergraph Laplacian.

Cardinality-based: Our proposed method reduces to the cardinality-based case when $\alpha = 0$. Indeed, it immediately follows from the definitions of EDVWs for both datasets that we effectively ignore the EDVWs by setting $\alpha = 0$, i.e., we assig a uniform value of 1 to vertices in a hyperedge. Hence, the corresponding submodular weight functions $w_e(\mathcal{S})$ depend only on the cardinality of $\mathcal{S}$.

[1]The implementation of our method and experiments can be found at https://github.com/bl166/hg_general_submodular_weights.

[2]https://trec.nist.gov/data/reuters/reuters.html

[3]https://archive.ics.uci.edu/ml/datasets/covertype

All-or-nothing: This is yet another special case of the proposed method when β approaches 0. From the definition of the splitting function in (10) it follows that there exists a small enough $\beta > 0$ under which the same penalty will be assigned to every possible cut of e.

Results. The evaluation of multiple settings is displayed in Fig. 1, including the two datasets (RCV1 and Covtype), two evaluation metrics (clustering error and NCC), as well as the performance metrics versus two parameters (α and β).

Performance versus parameter α: In Fig. 1a and 1e, the clustering errors are computed as the fraction of incorrectly clustered samples and plotted as a function of parameter α (fixing $\beta = 0.17$). Correspondingly, the NCC metric is shown in Fig. 1b and 1f. Focusing first on the RCV1 dataset (Fig. 1a-1b), it can be observed that the proposed method always yields smaller or same metrics (both clustering error and NCC) compared to the random walk-based baseline using the classical Laplacian. Moreover, the cardinality-based baseline ($\alpha = 0$) deteriorates the performance of the general case ($\alpha > 0$), which highlights the modeling flexibility as a major advantage of the proposed method. Turning to the Covtype dataset (Fig. 1e-1f), the trends are very similar. More conspicuously, greater performance gaps appear between our method and the random walk-based method, underscoring the importance of utilizing the non-linear 1-Laplacian in spectral clustering.

Performance versus parameter β: In Fig. 1c, 1d, 1g, and 1h, the clustering errors and NCC values are plotted against different choices of the splitting function parameter β (fixing $\alpha = 2$). The proposed method significantly outperforms the random walk-based baseline, and its performance gradually improves as it departs from all-or-nothing (very small β) into the general submodular weights scenario.

V. Conclusions

We presented an implementation of the 1-spectral clustering algorithm for hypergraphs with generic submodular splitting functions. There are several directions for future work: (1) Design multiway partitioning algorithms based on nonlinear Laplacians; (2) Explore applications of p-Laplacians for values of p that go beyond $p = 1$ and $p = 2$; and (3) Tackle the open problem of whether all submodular splitting functions are graph reducible.

References

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

[2] S. Amghibech, "Eigenvalues of the discrete p-laplacian for graphs," *Ars Combinatoria*, vol. 67, pp. 283–302, 2003.

[3] T. Bühler and M. Hein, "Spectral clustering based on the graph p-laplacian," in *International Conference on Machine Learning*, pp. 81–88, 2009.

[4] A. Szlam and X. Bresson, "Total variation and cheeger cuts," in *International Conference on Machine Learning*, pp. 1039–1046, 2010.

[5] A. R. Benson, D. F. Gleich, and J. Leskovec, "Higher-order organization of complex networks," *Science*, vol. 353, no. 6295, pp. 163–166, 2016.

[6] M. T. Schaub, Y. Zhu, J.-B. Seby, T. M. Roddenberry, and S. Segarra, "Signal processing on higher-order networks: Livin'on the edge... and beyond," *Signal Processing*, vol. 187, p. 108149, 2021.

[7] J. Li, J. He, and Y. Zhu, "E-tail product return prediction via hypergraph-based local graph cut," in *International Conference on Knowledge Discovery & Data Mining*, pp. 519–527, 2018.

[8] K. Hayashi, S. G. Aksoy, C. H. Park, and H. Park, "Hypergraph random walks, laplacians, and clustering," in *Conference on Information and Knowledge Management*, pp. 495–504, 2020.

[9] Y. Zhu, B. Li, and S. Segarra, "Co-clustering vertices and hyperedges via spectral hypergraph partitioning," in *European Signal Processing Conference*, pp. 1416–1420, 2021.

[10] U. Chitra and B. Raphael, "Random walks on hypergraphs with edge-dependent vertex weights," in *International Conference on Machine Learning*, pp. 1172–1181, 2019.

[11] N. Veldt, A. R. Benson, and J. Kleinberg, "Hypergraph cuts with general splitting functions," *SIAM Review*, vol. 64, no. 3, pp. 650–685, 2022.

[12] P. Li and O. Milenkovic, "Submodular hypergraphs: p-laplacians, cheeger inequalities and spectral clustering," in *International Conference on Machine Learning*, pp. 3014–3023, 2018.

[13] Y. Zhu and S. Segarra, "Hypergraph cuts with edge-dependent vertex weights," *Applied Network Science*, vol. 7, no. 1, p. 45, 2022.

[14] Y. Zhu and S. Segarra, "Hypergraphs with edge-dependent vertex weights: p-laplacians and spectral clustering," *arXiv preprint arXiv:2208.07457*, 2022.

[15] Y. Zhu, B. Li, and S. Segarra, "Hypergraphs with edge-dependent vertex weights: Spectral clustering based on the 1-laplacian," in *International Conference on Acoustics, Speech and Signal Processing*, pp. 8837–8841, 2022.

[16] M. Hein and T. Bühler, "An inverse power method for nonlinear eigenproblems with applications in 1-spectral clustering and sparse PCA," *Advances in Neural Information Processing Systems*, vol. 23, 2010.

[17] A. Ene and H. Nguyen, "Random coordinate descent methods for minimizing decomposable submodular functions," in *International Conference on Machine Learning*, pp. 787–795, 2015.

[18] P. Wolfe, "Finding the nearest point in a polytope," *Mathematical Programming*, vol. 11, no. 1, pp. 128–149, 1976.

[19] S. Fujishige, "Submodular systems and related topics," in *Mathematical Programming at Oberwolfach II*, pp. 113–131, 1984.

[20] D. Chakrabarty, P. Jain, and P. Kothari, "Provable submodular minimization via fujishige-wolfe algorithm," *Advances in Neural Information Processing Systems*, 2014.

[21] L. Lovász, "Submodular functions and convexity," in *Mathematical programming The state of the art*, pp. 235–257, 1983.

[22] F. R. Chung, *Spectral Graph Theory*, vol. 92. American Mathematical Soc.

[23] F. Tudisco and M. Hein, "A nodal domain theorem and a higher-order cheeger inequality for the graph p-laplacian," *Journal of Spectral Theory*, vol. 8, no. 3, pp. 883–908, 2018.

[24] G. Carlsson, F. Mémoli, A. Ribeiro, and S. Segarra, "Hierarchical clustering of asymmetric networks," *Adv. Data Anal. Classif.*, vol. 12, pp. 65–105, 2018.

[25] D. Wagner and F. Wagner, "Between min cut and graph bisection," in *International Symposium on Mathematical Foundations of Computer Science*, pp. 744–750, 1993.

[26] S. Jegelka, F. Bach, and S. Sra, "Reflection methods for user-friendly submodular optimization," *Advances in Neural Information Processing Systems*, vol. 26, 2013.

[27] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *SIAM Journal on Imaging Sciences*, vol. 2, no. 1, pp. 183–202, 2009.

[28] Y. E. Nesterov, "A method for solving the convex programming problem with convergence rate o (1/kˆ 2)," in *Dokl. akad. nauk Sssr*, vol. 269, pp. 543–547, 1983.

[29] A. Chambolle and T. Pock, "A first-order primal-dual algorithm for convex problems with applications to imaging," *Journal of mathematical imaging and vision*, vol. 40, no. 1, pp. 120–145, 2011.

[30] F. Tudisco, P. Mercado, and M. Hein, "Community detection in networks via nonlinear modularity eigenvectors," *SIAM Journal on Applied Mathematics*, vol. 78, no. 5, pp. 2393–2419, 2018.

[31] D. D. Lewis, Y. Yang, T. Russell-Rose, and F. Li, "RCV1: A new benchmark collection for text categorization research," *Journal of Machine Learning Research*, vol. 5, pp. 361–397, 2004.

[32] J. Leskovec, A. Rajaraman, and J. D. Ullman, *Mining of massive data sets*. 2020.