ECAI 2023
K. Gal et al. (Eds.)

doi:10.3233/FAIA230416

Spectral Normalized-Cut Graph Partitioning with Fairness Constraints

Jia Li[a], Yanhao Wang[a;*] and Arpit Merchant[b]

[a]School of Data Science and Engineering, East China Normal University, Shanghai, China
[b]Department of Computer Science, University of Helsinki, Helsinki, Finland
ORCiD ID: Yanhao Wang https://orcid.org/0000-0002-7661-3917,
Arpit Merchant https://orcid.org/0000-0001-8143-1539

Abstract. Normalized-cut graph partitioning aims to divide the set of nodes in a graph into k disjoint clusters to minimize the fraction of the total edges between any cluster and all other clusters. In this paper, we consider a fair variant of the partitioning problem wherein nodes are characterized by a categorical sensitive attribute (e.g., *gender* or *race*) indicating membership to different demographic groups. Our goal is to ensure that each group is approximately proportionally represented in each cluster while minimizing the normalized cut value. To resolve this problem, we propose a two-phase spectral algorithm called FNM. In the first phase, we add an augmented Lagrangian term based on our fairness criteria to the objective function for obtaining a fairer spectral node embedding. Then, in the second phase, we design a rounding scheme to produce k clusters from the fair embedding that effectively trades off fairness and partition quality. Through comprehensive experiments on nine benchmark datasets, we demonstrate the superior performance of FNM compared with three baseline methods.

1 Introduction

Machine learning algorithms are widely used to make decisions that can directly affect people's lives in various domains, including banking [22], healthcare [11], education [41], and criminal justice [9], to name a few. However, a large body of work [18, 42] has indicated that these algorithms, if left unchecked, often present discriminatory outcomes for particular demographic groups. To address such concerns, recent studies have incorporated different notions of fairness into unsupervised learning problems [12, 16, 33, 39]. In particular, Chierichetti *et al.* [16] pioneered *fair clustering* for a set of points represented as vectors in Euclidean space and further characterized by a categorical sensitive attribute (e.g., *gender* or *race*) indicating membership in a demographic group (e.g., *female* or *Asian*). In addition to minimizing the typical clustering objective, the problem also requires that the proportion of each demographic group within each cluster is roughly the same as its proportion in the dataset population (referred to as *proportional fairness*). Beyond clustering, however, fairness in the partitioning of graphs is relatively under-explored despite its broad applications to community detection [15, 44, 54] and computer vision [14, 60].

In this paper, we aim to fill this gap by defining a fair version of the normalized-cut graph partitioning problem [50, 59]. Informally, for a graph $G = (V, E)$, the original partitioning objective is to divide the set V of nodes into k disjoint clusters such that the fractions of inter-cluster edges are minimized while the fractions of intra-cluster edges are maximized. Such an objective is measured by the normalized cut (Ncut) value. In our fair variant, we further assume that each node belongs to one of m sensitive groups and consider the notion of *range-based proportional fairness* [7] generalized from that in [16]. Specifically, this requires that, in each of the k clusters, the proportion of nodes of any group $c \in \{1, 2, \dots, m\}$ is at least β_c (lower bound) and at most α_c (upper bound) for two parameters $\beta_c, \alpha_c \in [0, 1]$. Our overall objective, thus, is to produce a k-partition that minimizes the Ncut value while also satisfying the above fairness constraint.

To the best of our knowledge, the most relevant algorithms to our problem are those for spectral clustering with group fairness constraints [34, 55] due to the inherent connection between spectral clustering and normalized-cut graph partitioning. But those algorithms suffer from two key limitations when applied to graph partitioning. First, although they incorporate a special case of the range-based fairness constraint with $\alpha_c = \beta_c, \forall c \in \{1, 2, \dots, m\}$ (i.e., the original *proportional fairness* in [16]) into spectral node embedding, they still consider running standard k-means [37] on node vectors to obtain a k-partition. Consequently, they cannot guarantee how close the partitioning is to satisfying the (original or range-based) fairness constraint. Second, they do not provide any tunable trade-off between fairness and partition quality.

Our Contributions. In this paper, we propose a novel algorithmic framework for fair normalized-cut graph partitioning that addresses the above two limitations. That is, we parameterize the desired level of range-based proportional fairness as a constraint to be satisfied and naturally trade off the Ncut value (i.e., *quality*) and proportionality (i.e., *fairness*) of the partitioning. Similar to [50, 59], we transform the problem of minimizing the Ncut value of a graph into an equivalent trace minimization problem on its Laplacian matrix. Generally, our algorithm, which we refer to as FNM, comprises two phases. In the first phase, we relax the original integer trace minimization problem to allow fractional memberships, add an augmented Lagrangian term [46] based on our fairness criteria to the objective function of the relaxed problem, and use the OptStiefelGBB method [58] to obtain a fairer embedding from which the partitioning found is closer to being fair. Then, in the second phase, we apply a novel rounding scheme, adapted from Lloyd's k-means clustering algorithm [37], to generate a fair partitioning from node vectors in the embedding

* Corresponding Author. Email: yhwang@dase.ecnu.edu.cn.

space. Specifically, we initialize the cluster centers, alternately assign vectors fairly to clusters, and update the centers for a fixed number of iterations or until the stopping condition is met. Each assignment step first solves a linear program to produce k nearly-fair clusters and then performs reassignments to construct strictly-fair clusters without significantly reducing partition quality.

Finally, we evaluate the performance of our FNM algorithm on nine benchmarking datasets ranging in size from 155 to 1.6M nodes along three metrics – partition quality, fairness, and time efficiency, compared to three competitive baselines and ten variants for ablation study. Our key findings are summarized below: *i*) FNM offers improved trade-offs between fairness and partition quality compared to the three baselines while scaling effectively to million-node-sized graphs; *ii*) our proposed fair embedding and rounding algorithms independently and jointly improve the quality of graph partitions with range-based proportional fairness constraints compared to general node embeddings and rounding schemes.

1.1 Other Related Works

Min-Cut Graph Partitioning. Partitioning nodes in a graph into disjoint subsets to minimize an objective function, such as *ratio cut* [28], *normalized cut* [50], and *Cheeger cut* [10], is a fundamental combinatorial optimization problem. Since all those graph partitioning problems are NP-hard [28, 50], different heuristic algorithms were designed for them, among which spectral methods [28, 29, 45, 50] have attracted the most attention. The basic idea of spectral methods is to relax the original integer minimization problems into continuous optimization problems, which are solved by computing the eigenvectors of the Laplacian matrix, and to find the partitioning using k-means or an alternative rounding method. Other relaxation-based methods [13, 31, 35] improved the efficiency over original spectral methods by avoiding eigendecomposition. However, none of them incorporate the notion of *fairness* into graph partitioning problems.

Fair Clustering. There has been rich literature on fair clustering algorithms. Chierichetti *et al.* [16] first introduced the notion of *proportional fairness* for clustering and then proposed fairlet decomposition algorithms for fair k-median and k-center clustering in the case of two groups. Following this work, the problem has been further generalized to handle more than two groups [30], permit lower and upper bounds on the fraction of points from a group [7,8,27], and allow probabilistic group memberships [21]. In addition, more efficient fair clustering algorithms have been proposed based on faster fairlet computation [5] or coresets [30,49]. Then, several studies [19,20,61] explored how to achieve better trade-offs between the fairness and clustering objectives. Other fair variants of clustering problems focused on different fairness notions, including individual-level fairness [39, 43, 53], fair center selection [33, 52], and minimax losses among groups [17, 24, 40]. However, all the above methods are primarily designed for i.i.d. data rather than graph data. They cannot be directly applied to graph partitioning because graphs are high-dimensional and sparse, and they will incur huge computational costs and often return inferior results due to the curse of dimensionality.

There were also a few studies on fairness in spectral methods and other graph clustering problems. In addition to [34, 55], Gupta and Dukkipati [26] further investigated spectral clustering with individual fairness constraints. Moreover, Ahmadian *et al.* [1], Friggstad and Mousavi [23], and Ahmadian and Negahbani [2] studied fair correlation clustering on signed graphs. Anagnostopoulos *et al.* [3] applied spectral methods to fair densest subgraph discovery on graphs. These algorithms are interesting but not comparable to our algorithm.

2 Problem Definition

Normalized-Cut Graph Partitioning. Define $[n] := \{1, 2, \dots, n\}$ for any positive integer n. Let $G = (V, E)$ be an undirected graph, where $V = [n]$ is the set of n nodes and $E \subseteq V \times V$ is the set of edges. We use $\boldsymbol{W} = (w_{ij})_{i,j\in[n]}$ to denote the adjacency matrix of G, where each entry $w_{ij} \geq 0$ is the weight of edge (i, j) (always equal to 1 for the unweighted case) if it exists or 0 otherwise. We consider $w_{ii} = 0$ for any $i \in [n]$. The degree matrix $\boldsymbol{D} = (d_i)_{i\in[n]}$ is a diagonal matrix with the degree $d_i = \sum_{j=1}^{n} w_{ij} \geq 0$ of each node i on its diagonal. Given an undirected graph G, an integer $k \geq 2$, we aim to find a partitioning $\mathcal{C} = \{C_1, \dots, C_k\}$ of V into k disjoint clusters, i.e., $\bigcup_{l=1}^{k} C_l = V$ and $C_l \cap C_{l'} = \emptyset$ for any $l \neq l' \in [k]$, to minimize the normalized cut [59] (Ncut) value as follows:

$$\mathsf{Ncut}(\mathcal{C}) := \sum_{l=1}^{k} \frac{\mathsf{cut}(C_l)}{\mathsf{vol}(C_l)} = \sum_{l=1}^{k} \frac{\sum_{i\in C_l, j\in V\setminus C_l} w_{ij}}{\sum_{i\in C_l, j\in V} w_{ij}}. \tag{1}$$

The Ncut minimization problem is well-known as NP-hard [50].

Fairness Constraint. In the fair variant of graph partitioning, we consider that the node set V consists of several demographic groups defined by a categorical sensitive attribute, e.g., gender or race. Formally, suppose that V is divided into m disjoint groups indexed by $[m]$, and an indicator function $\phi : [n] \mapsto [m]$ maps each node $i \in [n]$ to the group $\phi(i)$ it belongs to. Let $V_c = \{i \in [n] : \phi(i) = c\}$ be the subset of nodes from group c in V. We assume that $\bigcup_{c=1}^{m} V_c = V$ and $V_c \cap V_{c'} = \emptyset, \forall c \neq c' \in [m]$. For ease of presentation, we denote the group membership as an indicator matrix $\boldsymbol{M} \in \{0, 1\}^{n\times m}$, where $\boldsymbol{M}_{i,c} = 1$ if $\phi(i) = c$ and 0 otherwise. We follow a notion of *range-based proportional fairness* in [7] to require that every demographic group is approximately proportionally represented in all the k clusters. We define the fairness constraint by two vectors $\boldsymbol{\alpha}, \boldsymbol{\beta} \in [0, 1]^m$ that specify the upper and lower bounds α_c, β_c on the percentage of nodes from group c. We say a partitioning $\mathcal{C}$ is $(\boldsymbol{\alpha}, \boldsymbol{\beta})$*-proportionally fair* if $\beta_c \leq \frac{|V_c\cap C_l|}{|C_l|} \leq \alpha_c$ for any C_l and V_c. In practice, we parameterize $\boldsymbol{\alpha}, \boldsymbol{\beta}$ by a fairness variable $\sigma \in [0, 1]$ as $\alpha_c = \min\{r_c/(1-\sigma), 1\}$ and $\beta_c = r_c\cdot(1-\sigma)$, where $r_c = |V_c|/n$.[1] For example, if the percentage of females is 60% in the population of all nodes, the fairness constraint requires that the percentage of females in each cluster should be between 48% and 75% when $\sigma = 0.2$. The value of σ can be interpreted as how loose the fairness constraint is, where $\sigma = 0$ corresponds to every group in each cluster having the same ratio as that group in the population, and $\sigma = 1$ corresponds to no fairness constraint at all. Given all the above notions, we formally define the normalized-cut graph partitioning problem under $(\boldsymbol{\alpha}, \boldsymbol{\beta})$-proportional fairness as follows.

Definition 1 *Given an undirected graph $G = (V, E)$, a set of m groups $V_1, \dots, V_m \subseteq V$, two fairness vectors $\boldsymbol{\alpha}, \boldsymbol{\beta} \in [0, 1]^m$, and an integer $k \geq 2$, find an $(\boldsymbol{\alpha}, \boldsymbol{\beta})$-proportionally fair partitioning $\mathcal{C} = \{C_1, \dots, C_k\}$ of V into k disjoint clusters such that* $\mathsf{Ncut}(\mathcal{C})$ *in Eq. 1 is minimized.*

The problem in Definition 1 is NP-hard since the vanilla Ncut minimization problem is its special case when $m = 1$. Next, we will approach the problem by extending the spectral Ncut minimization algorithm. Note that our method can be adapted to *ratio cut* [28] and any other cut measure with an equivalent spectral formulation. This paper focuses on Ncut due to its prevalence and space limitations.

[1] Any other parameterization scheme (e.g., $\beta_c = r_c \cdot (1 - \sigma)$ and $\alpha_c = \beta_c + \sigma$) is also compatible with our formulation as long as it guarantees that $\beta_c \leq \alpha_c$ and $\beta_c, \alpha_c \in [0, 1]$ for any $c \in [m]$.

3 Our Algorithm

We now introduce our two-phase spectral algorithm, FNM, for the problem of normalized-cut minimization with range-based proportional fairness constraints. Next, we will present our range-based fair spectral embedding and rounding methods in Sections 3.1 and 3.2.

3.1 Range-based Fair Spectral Embedding

(Unconstrained) Spectral Normalized-Cut Minimization. We begin with a review of (unconstrained) spectral normalized-cut minimization. According to [59], the Ncut value can be expressed in terms of the graph Laplacian $\boldsymbol{L} := \boldsymbol{D} - \boldsymbol{W}$ and a cluster membership indicator matrix $\boldsymbol{H} \in \mathbb{R}^{n \times k}$ as $\mathsf{Ncut}(\mathcal{C}) = \mathsf{trace}(\boldsymbol{H}^\top \boldsymbol{L}\boldsymbol{H})$, where

$$\boldsymbol{H}_{i,l} = \begin{cases} \frac{1}{\sqrt{\mathsf{vol}(C_l)}}, & \text{if } i \in C_l; \\ 0, & \text{otherwise;} \end{cases} \quad \forall i \in [n], \forall l \in [k]. \tag{2}$$

As such, the Ncut minimization problem is equivalent to minimizing $\mathsf{trace}(\boldsymbol{H}^\top \boldsymbol{L}\boldsymbol{H})$ over all possible $\boldsymbol{H}$ in the form of Eq. 2. However, the transformed problem is still NP-hard due to its combinatorial nature. Therefore, the (normalized) spectral method [59] solves the following relaxed continuous optimization problem by allowing fractional assignments of nodes to clusters:

$$\min_{\boldsymbol{H} \in \mathbb{R}^{n \times k}} \mathsf{trace}(\boldsymbol{H}^\top \boldsymbol{L}\boldsymbol{H}) \;\text{ s.t. }\; \boldsymbol{H}^\top \boldsymbol{D}\boldsymbol{H} = \boldsymbol{I}_k, \tag{3}$$

where $\boldsymbol{I}_k$ is an identity matrix of size $k \times k$. Note that $\boldsymbol{H}$ in the form of Eq. 2 must satisfy $\boldsymbol{H}^\top \boldsymbol{D}\boldsymbol{H} = \boldsymbol{I}_k$. To solve the problem in Eq. 3, under an assumption that $d_i > 0, \forall i \in [n]$ (i.e., G has no isolated node), we substitute $\boldsymbol{H}$ with $\boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{T}$ as follows:

$$\min_{\boldsymbol{T} \in \mathbb{R}^{n \times k}} \mathsf{trace}(\boldsymbol{T}^\top \boldsymbol{D}^{-\frac{1}{2}} \boldsymbol{L} \boldsymbol{D}^{-\frac{1}{2}} \boldsymbol{T}) \;\text{ s.t. }\; \boldsymbol{T}^\top \boldsymbol{T} = \boldsymbol{I}_k. \tag{4}$$

By Rayleigh-Ritz theorem [38, § 5.2.2], an optimal solution to the problem in Eq. 4 is the matrix $\boldsymbol{T}$ which has the eigenvectors of $\boldsymbol{D}^{-\frac{1}{2}} \boldsymbol{L} \boldsymbol{D}^{-\frac{1}{2}}$ with respect to its k smallest eigenvalues as columns.

Range-based Fair Spectral Ncut Minimization. We incorporate range-based proportional fairness into the problem in Eq. 4. Let us define two matrices $\boldsymbol{A} = [\boldsymbol{\alpha}, \ldots, \boldsymbol{\alpha}]^\top, \boldsymbol{B} = [\boldsymbol{\beta}, \ldots, \boldsymbol{\beta}]^\top \in \mathbb{R}^{n \times m}$ with the two fairness vectors $\boldsymbol{\alpha}, \boldsymbol{\beta} \in [0,1]^m$ as their rows. By definition, a partitioning $\mathcal{C} = \{C_1, \ldots, C_k\}$ denoted as $\boldsymbol{H}$ in the form of Eq. 2 is fair if and only if $(\boldsymbol{A} - \boldsymbol{M})^\top \boldsymbol{H} \geq \boldsymbol{0}$ and $(\boldsymbol{M} - \boldsymbol{B})^\top \boldsymbol{H} \geq \boldsymbol{0}$, where $\boldsymbol{0}$ is the zero matrix of size $m \times k$, because $\beta_c \cdot |C_l| \leq |V_c \cap C_l| \leq \alpha_c \cdot |C_l|$ if and only if the (c,j)-th entries of $(\boldsymbol{M} - \boldsymbol{B})^\top \boldsymbol{H}$ and $(\boldsymbol{A} - \boldsymbol{M})^\top \boldsymbol{H}$ are nonnegative for any $c \in [m]$ and $l \in [k]$. In this way, the fairness constraints are expressed equivalently as *linear constraints* in matrix form.

Given the above result, the problem in Definition 1 is transformed to minimizing $\mathsf{trace}(\boldsymbol{H}^\top \boldsymbol{L}\boldsymbol{H})$ over all $\boldsymbol{H}$ in the form of Eq. 2 with two additional linear constraints of $(\boldsymbol{A} - \boldsymbol{M})^\top \boldsymbol{H} \geq \boldsymbol{0}$ and $(\boldsymbol{M} - \boldsymbol{B})^\top \boldsymbol{H} \geq \boldsymbol{0}$. By applying the same relaxation procedure as for the unconstrained problem, we obtain the following relaxed problem for range-based fair spectral Ncut minimization:

$$\min_{\boldsymbol{T} \in \mathbb{R}^{n \times k}} \quad \mathsf{trace}(\boldsymbol{T}^\top \boldsymbol{D}^{-\frac{1}{2}} \boldsymbol{L} \boldsymbol{D}^{-\frac{1}{2}} \boldsymbol{T}) \tag{5a}$$

$$\text{subject to} \quad \boldsymbol{T}^\top \boldsymbol{T} = \boldsymbol{I}_k \tag{5b}$$

$$(\boldsymbol{A} - \boldsymbol{M})^\top \boldsymbol{D}^{-\frac{1}{2}} \boldsymbol{T} \geq \boldsymbol{0} \tag{5c}$$

$$(\boldsymbol{M} - \boldsymbol{B})^\top \boldsymbol{D}^{-\frac{1}{2}} \boldsymbol{T} \geq \boldsymbol{0} \tag{5d}$$

Algorithm 1 Range-based Fair Spectral Embedding

Input: Graph G with adjacency matrix $\boldsymbol{W} \in \mathbb{R}^{n \times n}$, group matrix $\boldsymbol{M} \in \mathbb{R}^{n \times m}$, fairness vectors $\boldsymbol{\alpha}, \boldsymbol{\beta} \in [0,1]^m$, an integer $k \geq 2$
Parameters: $T_1, \boldsymbol{\Lambda}_0, \mu_0, \xi, \varepsilon_1; T_2, \tau, \varepsilon_2$
Output: Embedding matrix $\boldsymbol{H} \in \mathbb{R}^{n \times k}$

```
Compute D, L and A, B based on W and α, β, respectively.
Initialize an arbitrary matrix T_0 ∈ R^{n×k} with T_0^⊤ T_0 = I_k.
for t = 0, 1, ..., T_1 do
    Formulate the problem in Eq. 7 with T_t, Λ_t, and μ_t.
    repeat
        Use Eq. 9 or 12 to compute T'_t w.r.t. T_t and τ.
        Set T_t ← T'_t and update τ using the Barzilai-Borwein
        method [6].
    until ||∇_{T_t} L_{μ_t}(T_t, Λ_t)||_F ≤ ε_2 or after T_2 iterations
    Set T_{t+1} ← T_t and update Λ_{t+1}, μ_{t+1} based on Eq. 8.
    if ||min{P(T_{t+1}), 0}||_F ≤ ε_1 then
        H ← D^{-1/2} T_{t+1} and break.
    end if
end for
return H
```

Unlike Eq. 4, the problem in Eq. 5 cannot be directly solved by eigendecomposition due to two additional constraints in Eqs. 5c and 5d.

Range-based Fair Spectral Embedding with Augmented Lagrangian Method and OptStiefelGBB. To resolve the problem in Eq. 5, we propose a novel algorithm based on the augmented Lagrangian method [46] for constrained optimization and OptStiefelGBB [58] for optimization with orthogonal constraints, as presented in Algorithm 1, to find a solution $\boldsymbol{T}$ and a matrix $\boldsymbol{H} = \boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{T}$ denoting a fractional assignment of each node in V to k clusters.

The basic idea of the augmented Lagrangian method is to solve a constrained optimization problem by converting the constraints into penalty and Lagrange multiplier terms in the objective function. In our problem, the violation of fairness constraints in Eqs. 5c and 5d by a matrix $\boldsymbol{T}$ is denoted as the following penalty matrix:

$$\boldsymbol{P}(\boldsymbol{T}) := [(\boldsymbol{A} - \boldsymbol{M})^\top \boldsymbol{D}^{-\frac{1}{2}} \boldsymbol{T}, (\boldsymbol{M} - \boldsymbol{B})^\top \boldsymbol{D}^{-\frac{1}{2}} \boldsymbol{T}] \in \mathbb{R}^{m \times 2k}.$$

Based on [46, § 17.4], the objective function in Eq. 5a with penalty and Lagrange multiplier terms is as follows:

$$L_\mu(\boldsymbol{T}, \boldsymbol{\Lambda}) := \mathsf{trace}(\boldsymbol{T}^\top \boldsymbol{D}^{-\frac{1}{2}} \boldsymbol{L} \boldsymbol{D}^{-\frac{1}{2}} \boldsymbol{T}) + \sum_{c=1}^{m} \sum_{l=1}^{2k} \rho(\boldsymbol{P}_{c,l}(\boldsymbol{T}), \boldsymbol{\Lambda}_{c,l}, \mu), \tag{6}$$

where $\mu > 0$ is the penalty parameter, $\boldsymbol{\Lambda}$ is the estimation of Lagrange multipliers, and ρ is defined as:

$$\rho(p, \lambda, \mu) := \begin{cases} -\lambda p + \frac{1}{2}\mu p^2, & \text{if } p - \frac{\lambda}{\mu} \leq 0; \\ -\frac{1}{2\mu}\lambda^2, & \text{otherwise.} \end{cases}$$

The augmented Lagrangian method starts from the initial parameters $\mu_0 > 0$, $\boldsymbol{\Lambda}_0 = \boldsymbol{0}_{m \times 2k}$ and a solution $\boldsymbol{T}_0 \in \mathbb{R}^{n \times k}$ with $\boldsymbol{T}_0^\top \boldsymbol{T}_0 = \boldsymbol{I}_k$. Then, it solves a sub-problem and iteratively updates the parameters. Here, the t-th sub-problem ($t = 0, 1, \ldots$) to solve is:

$$\min_{\boldsymbol{T} \in R^{n \times k}} L_{\mu_t}(\boldsymbol{T}, \boldsymbol{\Lambda}_t) \;\text{ s.t. }\; \boldsymbol{T}^\top \boldsymbol{T} = \boldsymbol{I}_k. \tag{7}$$

By solving Eq. 7, it obtains a new solution $\boldsymbol{T}_{t+1}$ and then updates μ and $\boldsymbol{\Lambda}$ as follows:

$$\boldsymbol{\Lambda}_{t+1} = \max\{\boldsymbol{\Lambda}_t - \mu_t \boldsymbol{P}(\boldsymbol{T}_{t+1}), \boldsymbol{0}\}, \quad \mu_{t+1} = \xi \mu_t, \tag{8}$$

where $\xi > 1$ is an amplification parameter. The augmented Lagrangian method terminates when the fairness violation of $\boldsymbol{T}_{t+1}$, defined as $\|\min\{\boldsymbol{P}(\boldsymbol{T}_{t+1}), \boldsymbol{0}\}\|_F$, is below an error parameter $\varepsilon_1 \geq 0$. After that, we finally obtain a "nearly fair" embedding matrix $\boldsymbol{H} = \boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{T}_{t+1}$ w.r.t. $\boldsymbol{T}_{t+1}$.

Then, we consider how to solve the sub-problem in Eq. 7 with OptStiefelGBB [58], a general method for optimization under orthogonality constraints. Its core idea is to model the feasible region as a (n, k)-Stiefel manifold [57], i.e., $\{\boldsymbol{T} \in \mathbb{R}^{n\times k} : \boldsymbol{T}^\top\boldsymbol{T} = \boldsymbol{I}_k\}$, and to apply Cayley transformation [56] to update the solution at each iteration. For the problem in Eq. 7, given a matrix $\boldsymbol{T} \in \mathbb{R}^{n\times k}$ with $\boldsymbol{T}^\top\boldsymbol{T} = \boldsymbol{I}_k$ and the gradients $\boldsymbol{\nabla_T} L_\mu(\boldsymbol{T}, \boldsymbol{\Lambda})$ ($\boldsymbol{\nabla_T}$ for short) of the augmented objective function $L_\mu(\boldsymbol{T}, \boldsymbol{\Lambda})$ w.r.t. $\boldsymbol{T}$, the updated matrix $\boldsymbol{T}'$ is expressed as $\boldsymbol{T} - \frac{\tau}{2}\boldsymbol{\Omega}(\boldsymbol{T} + \boldsymbol{T}')$, where $\tau > 0$ is the step size and $\boldsymbol{\Omega} = \boldsymbol{\nabla_T}\boldsymbol{T}^\top - \boldsymbol{T}\boldsymbol{\nabla}_{\boldsymbol{T}}^\top$. By applying Cayley transformation [56] on $\boldsymbol{\Omega}$, $\boldsymbol{T}'$ has a closed form expression as:

$$\boldsymbol{T}' = (\boldsymbol{I}_n + \frac{\tau}{2}\boldsymbol{\Omega})^{-1}(\boldsymbol{I}_n - \frac{\tau}{2}\boldsymbol{\Omega})\boldsymbol{T}. \tag{9}$$

Then, we compute the gradients using the chain rule as follows:

$$\boldsymbol{\nabla_T} = 2\boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{L}\boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{T} + \boldsymbol{A}'\boldsymbol{\nabla\rho}^{[0,k]} + \boldsymbol{B}'\boldsymbol{\nabla\rho}^{[k,2k]}, \tag{10}$$

where $\boldsymbol{A}' = ((\boldsymbol{A}-\boldsymbol{M})^\top\boldsymbol{D}^{-\frac{1}{2}})^\top$, $\boldsymbol{B}' = ((\boldsymbol{M}-\boldsymbol{B})^\top\boldsymbol{D}^{-\frac{1}{2}})^\top$, and $\boldsymbol{\nabla\rho}^{[0,k]}$, $\boldsymbol{\nabla\rho}^{[k,2k]}$ are the first and last k columns of $\boldsymbol{\nabla\rho} \in \mathbb{R}^{m\times 2k}$, respectively. The (c, l)-entry of $\boldsymbol{\nabla\rho}$ is expressed as

$$(\boldsymbol{\nabla\rho})_{c,l} = \begin{cases} -\boldsymbol{\Lambda}_{c,l} + \mu\boldsymbol{P}_{c,l}(\boldsymbol{T}), & \text{if } \boldsymbol{P}_{c,l} - \frac{\boldsymbol{\Lambda}_{c,l}}{\mu} \leq 0; \\ 0, & \text{otherwise.} \end{cases} \tag{11}$$

Since computing the inversion of the matrix $\boldsymbol{I}_n + \frac{\tau}{2}\boldsymbol{\Omega} \in \mathbb{R}^{n\times n}$ is time-consuming, the Sherman-Morrison-Woodbury formula [46, Appendix A] is further applied to devise a much more efficient update scheme when $k \ll n$ that only computes the inversion of a much smaller matrix $\boldsymbol{I}_{2k} + \frac{\tau}{2}\boldsymbol{Y}^\top\boldsymbol{X} \in \mathbb{R}^{2k\times 2k}$ as follows:

$$\boldsymbol{T}' = \boldsymbol{T} - \tau\boldsymbol{X}(\boldsymbol{I}_{2k} + \frac{\tau}{2}\boldsymbol{Y}^\top\boldsymbol{X})^{-1}\boldsymbol{Y}^\top\boldsymbol{T} \tag{12}$$

where $\boldsymbol{X} = [\boldsymbol{\nabla_T}, \boldsymbol{T}] \in \mathbb{R}^{n\times 2k}$ and $\boldsymbol{Y} = [\boldsymbol{T}, -\boldsymbol{\nabla_T}] \in \mathbb{R}^{n\times 2k}$. According to [58], $\boldsymbol{T}'$ by Eqs. 9 and 12 has two important properties: (*i*) $\boldsymbol{T}'^\top\boldsymbol{T}' = \boldsymbol{I}_k$ if $\boldsymbol{T}^\top\boldsymbol{T} = \boldsymbol{I}_k$; (*ii*) $L_\mu(\boldsymbol{T}', \boldsymbol{\Lambda}) \leq L_\mu(\boldsymbol{T}, \boldsymbol{\Lambda})$. Thus, with a proper step size τ, OptStiefelGBB always converges to a feasible stationary point after sufficient iterations. Following [58], we use the Barzilai-Borwein method [6] to adaptively adjust the step size τ at each iteration.

Time Complexity. Let T_1 and T_2 denote the maximum numbers of iterations in the augmented Lagrangian method and OptStiefelGBB, respectively. Computing the gradients $\boldsymbol{\nabla_T}$ for $L_\mu(\boldsymbol{T}, \boldsymbol{\Lambda})$ w.r.t. $\boldsymbol{T}$ takes $O\left((|E| + nm)k\right)$ time. Updating $\boldsymbol{T}$ with Eq. 9 or 12 needs $O(n^3)$ or $O(nk^2)$ time, respectively. As $m, k \leq n$ and $|E| \leq n^2$, the time complexity of Algorithm 1 is $O(T_1T_2n^3)$. When $k \ll n$, $m = O(k)$, and $|E| = O(n)$, the time complexity of Algorithm 1 is reduced to $O(T_1T_2nk^2)$.

3.2 Range-based Fair Rounding

Like vanilla spectral methods, the output $\boldsymbol{H}$ of Algorithm 1 is a k-dimensional node embedding matrix where the i-th row vector $\boldsymbol{h}_i$ represents a fractional assignment of node $i \in [n]$ to k clusters. Thus, we must round the fractional solution into an integral one for partitioning. However, a k-means clustering on embedding vectors, the standard rounding technique for spectral methods, is infeasible for the fair variant because the produced clusters may not be fair.

Next, we propose a novel rounding algorithm to produce a strictly fair partitioning scheme in Algorithm 2. Generally, it follows the same procedure as Lloyd's k-means clustering algorithm [37], which initializes k cluster centers, assigns each vector to one of the k centers to generate the clusters, and updates each center to the median of each generated cluster iteratively until the stopping condition is met. The difference from Lloyd's algorithm is that it requires the generated clusters at every iteration to be $(\boldsymbol{\alpha}, \boldsymbol{\beta})$-proportionally fair.

Nearly-Fair Initial Assignment via LP. Our rounding algorithm begins with running k-means++ [4] on the set $H = \{\boldsymbol{h}_i \in \mathbb{R}^k : i \in [n]\}$ of embedding vectors[2] to obtain an initial set $Q = \{\boldsymbol{q}_1, \ldots, \boldsymbol{q}_k\}$ of centers at the first iteration. Then, we formulate the following *fair assignment* problem [7] to assign the vectors in H to Q to minimize the l_2-loss while ensuring that the cluster around each center is $(\boldsymbol{\alpha}, \boldsymbol{\beta})$-proportionally fair:

Definition 2 *Given a point set H with m disjoint groups $H_1, \ldots, H_m$, a set Q of k centers, and two fairness vectors $\boldsymbol{\alpha}, \boldsymbol{\beta} \in [0, 1]^m$, find an assignment $\varphi : H \to Q$ that minimizes $\sum_{\boldsymbol{h}\in H}\|\boldsymbol{h} - \varphi(\boldsymbol{h})\|_2$ and ensures that $\beta_c \cdot |C_l| \leq |H_c \cap C_l| \leq \alpha_c \cdot |C_l|, \forall c \in [m], l \in [k]$, where $C_l = \{\boldsymbol{h} \in H : \varphi(\boldsymbol{h}) = \boldsymbol{q}_l\}$.*

By denoting an assignment as an indicator matrix $\boldsymbol{S} \in \{0, 1\}^{n\times k}$, where $\boldsymbol{S}_{i,l} = 1$ if $\varphi(\boldsymbol{h}_i) = \boldsymbol{q}_l$ and 0 otherwise, the problem in Definition 2 is represented as the following integer program (IP1):

$$\begin{aligned} \text{IP1} := \min \quad & \mathsf{trace}(\boldsymbol{C}^\top\boldsymbol{S}) & \text{(13a)} \\ \text{subject to} \quad & (\boldsymbol{A} - \boldsymbol{M})^\top\boldsymbol{S} \geq \boldsymbol{0} & \text{(13b)} \\ & (\boldsymbol{M} - \boldsymbol{B})^\top\boldsymbol{S} \geq \boldsymbol{0} & \text{(13c)} \\ & \boldsymbol{S}\boldsymbol{1}_k = \boldsymbol{1}_n,\ \boldsymbol{1}_n^\top\boldsymbol{S} \geq \boldsymbol{1}_k^\top & \text{(13d)} \\ & \boldsymbol{S} \in \{0, 1\}^{n\times k}, & \text{(13e)} \end{aligned}$$

where Eq. 13a denotes the minimization of the l_2-loss by setting the cost matrix $\boldsymbol{C} \in \mathbb{R}^{n\times k}$ with $\boldsymbol{C}_{i,l} = \|\boldsymbol{h}_i - \boldsymbol{q}_l\|_2$, Eqs. 13b and 13c represent the fairness conditions, and the two constraints in Eq. 13d mean that each vector must be assigned to exactly one center, and each center must be assigned with at least one vector. Since IP1 is NP-hard [51], we relax it to a linear program (LP1) by substituting the condition of Eq. 13e with $\boldsymbol{S} \in [0, 1]^{n\times k}$. After obtaining the optimal solution $\boldsymbol{S}^*$ to LP1 using any LP solver, we assign each vector $\boldsymbol{h}_i$ to the center $\varphi(\boldsymbol{h}_i) = \boldsymbol{q}_{l^*}$ with $l^* = \arg\max_{l\in[k]} \boldsymbol{S}^*_{i,l}$.

Reassignments to Generate Strictly Fair Clusters. Although the above assignment scheme can produce fairer clusters than standard k-means without fairness constraints, it may still violate the fairness conditions after rounding a fractional solution by LP1 to an integral one. Therefore, we must reassign some vectors (nodes) to other centers to produce a strictly fair partitioning scheme. To reduce the quality loss led by reassignments, we should (*i*) move as few nodes as possible and (*ii*) find the node leading to the smallest Ncut growth at each reassignment. For (*i*), we should seek a fair assignment φ' closest to the current assignment φ. Given an assignment φ, we use a matrix $\boldsymbol{N} = (n_{cl})_{c\in[m],l\in[k]} \in \mathbb{R}^{m\times k}$, where $n_{cl} = |H_c \cap C_l|$, to denote the number of nodes from each of the m groups in k clusters. Then, we define the problem of computing an optimal scheme with the least number of reassigned nodes in the following integer

[2] The nodes in V and vectors in H, as well as the groups defined on V and H, will be used interchangeably in this subsection.

Algorithm 2 Range-based Fair Rounding

Input: Embedded vectors $H \subseteq \mathbb{R}^k$ (resp. V), group indicator ϕ on H and V, fairness vectors $\boldsymbol{\alpha}, \boldsymbol{\beta} \in [0,1]^m$, an integer $k \geq 2$
Parameters: T_3, ε_3
Output: Partitioning $\mathcal{C}^* = \{C_1^*, \ldots, C_k^*\}$

Initialize $\mathcal{C}^* = \emptyset$ and $t = 0$.
Run k-means++ on H to obtain an initial set Q of centers.
repeat
  Set $t \leftarrow t + 1$.
  Compute $\boldsymbol{C}$ w.r.t. H, Q and solve LP1 to obtain $\boldsymbol{S}^*$.
  Assign each vector $\boldsymbol{h}_i \in H$ to a center $\varphi(\boldsymbol{h}_i) = \boldsymbol{q}_{l^*} \in Q$, where $l^* = \arg\max_{l \in [k]} \boldsymbol{S}^*_{i,l}$.
  Solve IP2 to obtain $\boldsymbol{N}'$ and compute $\boldsymbol{\Delta} = \boldsymbol{N}' - \boldsymbol{N}$.
  while there is any $c \in [m]$ and $l \in [k]$ with $\Delta_{cl} \neq 0$ **do**
    Pick arbitrary $c, C_l, C_{l'}$ with $\Delta_{cl} < 0$ and $\Delta_{cl'} > 0$.
    Let $\mathsf{cand} = \{i \in [n] : \phi(i) = c \wedge \varphi(\boldsymbol{h}_i) = \boldsymbol{q}_l\}$.
    Set $i^* \leftarrow \arg\min_{i \in \mathsf{cand}} \delta_{l,l'}(i)$ and $\varphi(\boldsymbol{h}_{i^*}) = \boldsymbol{q}_{l'}$.
    Update $\boldsymbol{N}$ and $\boldsymbol{\Delta}$ for the reassignment of i^*.
  end while
  Generate a partitioning $\mathcal{C} = \{C_1, \ldots, C_k\}$ from φ, where $C_l = \{i \in [n] : \varphi(\boldsymbol{h}_i) = \boldsymbol{q}_l\}$ for each $l \in [k]$.
  if $\mathcal{C}^* = \emptyset$ or $\mathsf{Ncut}(\mathcal{C}^*) > \mathsf{Ncut}(\mathcal{C})$ **then**
    $\mathcal{C}^* \leftarrow \mathcal{C}$.
  end if
  Set $Q' \leftarrow Q$ and $Q \leftarrow \{\boldsymbol{q}_l = \frac{1}{|C_l|}\sum_{i \in C_l} \boldsymbol{h}_i : l \in [k]\}$.
until $\sum_{l=1}^{k} \|\boldsymbol{q}_l - \boldsymbol{q}'_l\| \leq \varepsilon_3$ for $\boldsymbol{q}_l \in Q, \boldsymbol{q}'_l \in Q'$ or $t \geq T_3$
return $\mathcal{C}^*$

program (IP2):

$$\text{IP2} := \min \quad \sum_{c=1}^{m}\sum_{l=1}^{k} |n'_{cl} - n_{cl}| \tag{14a}$$

$$\text{subject to} \quad \beta_c n'_l \leq n'_{cl} \leq \alpha_c n'_l, \forall c \in [m], l \in [k] \tag{14b}$$

$$\sum_{l=1}^{k} n'_{cl} - n_{cl} = 0, \forall c \in [m] \tag{14c}$$

$$n'_{cl} \in \mathbb{Z}^+, \forall c \in [m], l \in [k] \tag{14d}$$

where n'_{cl} is the number of nodes in C_l from H_c after reassignments, $n'_l = \sum_{c=1}^{m} n'_{cl}$, and the objective value is twice as many as the number of reassignments from φ to fair φ'. To solve IP2, we can call an exact IP solver to find its optimal solution, which may take exponential time in the worst case but is still efficient in practice because the values of k and m are pretty small or run a heuristic search method, e.g., hill-climbing, to obtain a near-optimal solution in polynomial time. Using either method, we can find a reassignment scheme denoted as $\boldsymbol{\Delta} = \boldsymbol{N}' - \boldsymbol{N}$, where $\Delta_{cl} = n'_{cl} - n_{cl}$ is greater than 0 if Δ_{cl} nodes from V_c should be moved to C_l, is smaller than 0 if $-\Delta_{cl}$ nodes from V_c should be moved from C_l, or is equal to 0 if no reassignment is needed. For (*ii*), we first select a pair of clusters C_l and $C_{l'}$ with $\Delta_{cl} < 0$ and $\Delta_{cl'} > 0$ for a specific c, which corresponds to the movement of a node in V_c from C_l to $C_{l'}$. As reassignments can reduce partition quality, we want to find the node leading to as small Ncut growth as possible. Since computing the Ncut value of a given partitioning from scratch is time-consuming and we only need to recalculate the changed parts (i.e., one node and two clusters) for a reassignment, we obtain the following equation to update the Ncut value incrementally by taking the difference between the Ncut values after and before reassigning a node i from C_l to $C_{l'}$ and eliminating all the unchanged terms:

Table 1. Statistics of datasets in the experiments.

Dataset	$\|V\|$	$\|E\|$	Sensitive Attribute	m
Facebook	155	1,412	gender	2
German	1,000	21,742	gender	2
SBM	1,000	57,156	–	5
DBLP	1,061	2,576	continent	3
LastFM	7,624	27,806	country	4
Deezer	28,281	92,752	gender	2
Credit	29,460	136,196	education	3
Pokec-A	1,097,077	10,792,894	age	4
Pokec-G	1,632,803	22,301,964	gender	2

$$\delta_{l,l'}(i) = \frac{\mathsf{cut}(C_l) - d_i + 2z_{il}}{\mathsf{vol}(C_l) - d_i} - \frac{\mathsf{cut}(C_l)}{\mathsf{vol}(C_l)} + \frac{\mathsf{cut}(C_{l'}) + d_i - 2z_{il'}}{\mathsf{vol}(C_{l'}) + d_i} - \frac{\mathsf{cut}(C_{l'})}{\mathsf{vol}(C_{l'})}, \tag{15}$$

where $z_{il} = \sum_{j \in C_l} w_{ij}$. For each reassignment, we compute $\delta_{l,l'}(i)$ based on Eq. 15 for each eligible node i (i.e., $i \in V_c \cap C_l$) and pick the node with the smallest $\delta_{l,l'}(i)$ accordingly. After that, we update $\boldsymbol{N}, \boldsymbol{\Delta}$ and select the next group and pair of clusters for reassignment. The above process terminates when $\boldsymbol{N} = \boldsymbol{N}'$ and all the clusters have been fair. We obtain a fair partitioning $\mathcal{C} = \{C_1, \ldots, C_k\}$ from the final assignment, based on which we obtain an updated set $Q = \{\boldsymbol{q}_1, \ldots, \boldsymbol{q}_k\}$ of centers where $\boldsymbol{q}_l = \frac{1}{|C_l|}\sum_{i \in C_l} \boldsymbol{h}_i$. After Q is updated, the above procedures, i.e., solving LP1 & IP2 and reassigning nodes for fairness, will be executed again to acquire a new fair partitioning. This iterative procedure will terminate until the set Q of centers does not change significantly between two iterations or the total number of iterations reaches a predefined threshold T_3. Finally, a fair partitioning with the smallest Ncut value among all iterations will be returned as the final solution $\mathcal{C}^*$.

Time Complexity. The k-means++ algorithm takes $O(T_0 n k^2)$ time, where T_0 is the number of iterations. At each iteration of Algorithm 2, computing $\boldsymbol{C}$ takes $O(nk^2)$ time. Using the interior point method [32], solving LP1 takes $O(n^{4.5}k^{4.5}m)$ time in the worst case. The hill-climbing search takes $O(nmk)$ time to solve IP2. Furthermore, the time to perform one reassignment is $O(n)$, and there are at most $O(n)$ reassignments. But unlike Lloyd's k-means clustering algorithm, Algorithm 2 does not guarantee convergence to a local optimum after sufficient iterations. If the convergence is not reached, it will stop and return the best partitioning found after T_3 iterations. Suppose that $T_0 = O(T_3)$, the overall time complexity of Algorithm 2 is $O(T_3 n^{4.5} k^{4.5} m)$.

4 Experiments

In this section, we perform extensive empirical evaluations of our FNM algorithm. We introduce our experimental setup in Section 4.1 and describe our results in Section 4.2.

4.1 Experimental Setup

Datasets. We use eight public real datasets with sensitive attributes and one synthetic dataset in the experiments. *Facebook*, *LastFM*, *Deezer*, *Pokec-A*, *Pokec-G* are all social networks; *DBLP* is a coauthor network; *German* and *Credit* are similarity graphs created from i.i.d. data; and *SBM* is generated from a stochastic block model with random groups. If a graph is disconnected, we will extract and use

Table 2. Performance of different algorithms for normalized-cut graph partitioning with $k = 5$ clusters. Cells in lighter and darker gray colors denote results satisfying looser ($\sigma = 0.8$) and tighter ($\sigma = 0.2$) fairness constraints, respectively. For the Ncut values, we highlight the best overall results on each dataset in **bold** font and underline the best fair result when $\sigma = 0.8$. FSC is marked by "–" when it does not provide any solution due to huge memory consumption for eigendecomposition on dense matrices.

Dataset	SC			FSC			sFSC			FNM ($\sigma = 0.8$)			FNM ($\sigma = 0.2$)		
	Ncut	Balance	Time (s)	Ncut	Balance	Time (s)	Ncut	Balance	Time (s)	Ncut	Balance	Time (s)	Ncut	Balance	Time (s)
Facebook	**1.378**	0.458	0.233	1.401	0.623	0.252	1.401	0.623	0.193	**1.378**	0.458	0.225	1.550	0.81	0.312
German	**1.433**	0.211	0.357	1.442	0.583	0.817	1.442	0.583	0.581	**1.433**	0.211	0.486	1.498	0.8	2.841
SBM	**2.542**	0.226	0.375	2.619	0.245	0.921	2.619	0.245	0.703	**2.542**	0.226	0.660	3.348	0.81	2.608
DBLP	**0.022**	0	0.317	0.024	0	0.568	0.024	0	0.435	<u>0.050</u>	0.2	0.658	0.269	0.8	0.698
LastFM	**0.119**	0	0.526	0.185	0	102.2	0.185	0	1.462	<u>0.265</u>	0.2	4.624	0.699	0.8	7.014
Deezer	**0.038**	0.406	2.989	–	–	–	0.040	0.406	8.246	**0.038**	0.406	6.465	0.216	0.81	9.120
Credit	0.035	0.738	9.097	–	–	–	0.035	0.665	41.28	**0.034**	0.748	10.74	0.049	0.8	12.33
Pokec-A	**0.077**	0	241.5	–	–	–	**0.077**	0	686.9	<u>0.450</u>	0.209	270.6	2.701	0.8	254.4
Pokec-G	**0.070**	0.141	445.8	–	–	–	**0.070**	0.141	1024	<u>0.128</u>	0.254	525.0	1.330	0.81	644.8

its largest connected component. Table 1 summarizes the statistics of all processed datasets. Detailed descriptions of the above datasets are provided in Appendix A of the full version of this paper [36].

Baselines. We compare our FNM algorithm with the following three baseline methods for graph partitioning: *(i)* spectral clustering (SC) [59], *(ii)* fair spectral clustering (FSC) [34], and *(iii)* scalable fair spectral clustering (sFSC) [55]. In the ablation study, we compare our range-based fair spectral embedding (rFSE) in Algorithm 1 with the following six node embeddings: spectral embedding (SE) [59], fair spectral embedding (FSE) [34], scalable fair spectral embedding (sFSE) [55], DeepWalk (DW) [47], Node2Vec (N2V) [25], and FairWalk (FW) [48]; and our range-based fair rounding (FR) in Algorithm 2 with four alternatives: k-means++ [4], k-means++ with reassignments (K+R), fair k-means (FK) [7], and solving IP1 directly (IP). Note that FR, FK, and IP do not work on *Pokec-A* and *Pokec-G* because the IP/LP solver fails to provide solutions to IP1 or LP1 in a reasonable time. Alternatively, we use K+R together with rFSE to obtain the FNM results on both datasets.

Parameter Settings. For FNM, $\boldsymbol{\alpha}, \boldsymbol{\beta}$ are parameterized by $\sigma \in [0, 1]$ as per Section 2. By default, we set $\sigma = 0.2$ and 0.8 (resp. the common 80%-rule) to generate tight and loose fairness constraints. For Algorithm 1, we set $T_1 = 100$, $\varepsilon_1 = 10^{-6}$, $T_2 = 2,000$, $\tau = 10^{-3}$, and $\varepsilon_2 = 10^{-3}$. We perform a grid search on $\xi \in \{2, 4, \ldots, 10\}$ and $\mu_0 \in \{10^{-4}, 10^{-2}, 10^{0}, 10^{2}\}$ and select the combination of ξ, μ_0 achieving the lowest objective value for each experiment. For Algorithm 2, we set $T_0 = 100$, $T_3 = 10$, and $\varepsilon_3 = 10^{-4}$ for all experiments. Further details of our parameter-tuning procedure are provided in Appendix B of the full version of this paper [36]. For the baselines, we use the default parameters or the recommended methods for parameter tuning as given in their original papers.

Evaluation Metrics. Each method is evaluated in three aspects. First, we measure partition quality by the Ncut value in Eq. 1. Second, we adopt the notion of *balance* in [7, 16] as the metric for fairness. Given a set $\mathcal{C} = \{C_1, \ldots, C_k\}$ of k clusters and a set $\{V_1, \ldots, V_m\}$ of m groups, the proportion of group c in cluster C_l is defined as $r_{cl} = |C_l \cap V_c|/|C_l|$. Then, the *balance* of $\mathcal{C}$ is defined by $\mathsf{balance}(\mathcal{C}) := \min_{c\in[m],l\in[k]} \min\{r_c/r_{cl}, r_{cl}/r_c\}$, where $r_c = |V_c|/n$. Higher *balance* implies that the partitioning scheme is closer to being proportionally fair. Balance also serves as an indicator of whether the fairness constraints parameterized by σ are satisfied because $\mathcal{C}$ is $(\boldsymbol{\alpha}, \boldsymbol{\beta})$-proportionally fair iff $\mathsf{balance}(\mathcal{C}) \geq 1 - \sigma$. Third, we use *CPU time* to evaluate the efficiency of each method.

Implementation. We implement FNM in Python 3 and use Gurobi Optimizer to solve LPs and IPs. For each baseline, we either use a standard implementation in the SciPy library or the implementation published by the original authors. The experiments were conducted on a desktop with an Intel Core i5-9500 processor @3.0GHz and 32GB RAM running Ubuntu 20.04. Our code and data are published at https://github.com/JiaLi2000/FNM.

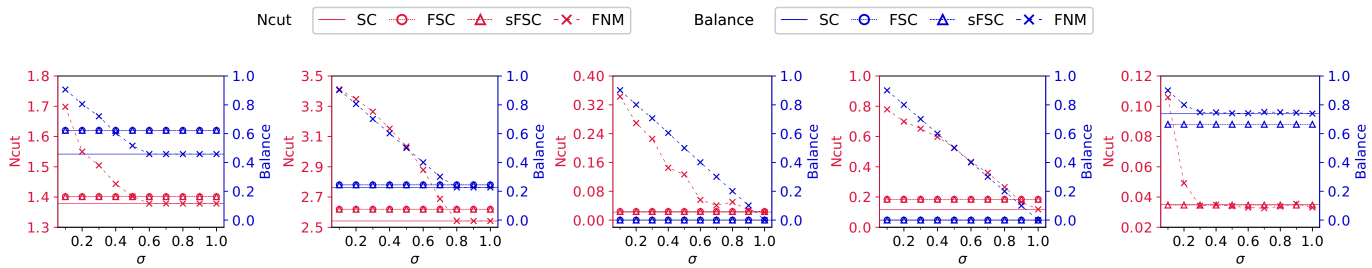
4.2 Experimental Results

Overview. Table 2 presents the performance of different algorithms for normalized-cut graph partitioning with two fairness constraints parameterized by $\sigma = 0.8$ and 0.2 when $k = 5$ on all nine datasets.

In terms of partition quality and fairness, the (unconstrained) SC mostly achieves the lowest Ncut values but fails to provide a fair partitioning when $\sigma = 0.8$ on four datasets while never meeting tighter fairness constraints when $\sigma = 0.2$. Although FSC and sFSC provide more balanced partitions than SC in some cases, they still cannot guarantee the satisfaction of fairness constraints. In addition, FSC does not return any results on medium and large graphs with over 10k nodes due to huge memory consumption for eigendecomposition on dense matrices. Next, we observe that FNM always provides fair partitioning schemes in all cases. If unconstrained SC returns fair solutions when $\sigma = 0.8$, FNM will achieve nearly the same Ncut values. Otherwise, the Ncut values of FNM will increase slightly to ensure fairness. Moreover, the Ncut values of FNM for $\sigma = 0.2$ are significantly higher than those for $\sigma = 0.8$, which can be regarded as the *price of fairness*.

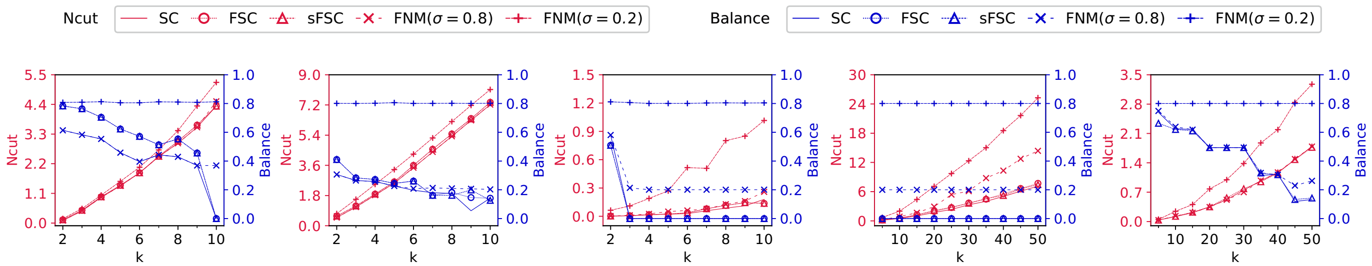
In terms of time efficiency, FNM runs slower than SC as it is more time-consuming in embedding and rounding. But FNM runs faster than FSC in most cases since it does not require eigendecomposition. Compared to sFSC, which improves the scalability of FSC by avoiding eigendecomposition on dense matrices, FNM runs slower on smaller graphs due to a longer time for fair rounding but becomes faster on larger graphs owing to the efficiency improvements in fair embedding. Finally, FNM is more efficient when $\sigma = 0.8$ than when $\sigma = 0.2$ due to fewer iterations for convergence.

Trade-off between Quality and Fairness. We present the performance of four algorithms with different fairness constraints parameterized by $\sigma = 0.1, 0.2, \ldots, 1$ in Figure 1. We ignore $\sigma = 0$ since no solution may exist for indivisibility. Since the results of SC, FSC, and sFSC are independent of σ, they are drawn as horizontal lines in the figure. For FNM, as the value of σ decreases (when the fairness constraints become looser), the Ncut and balance values also decrease. When $\sigma = 1$ (no fairness constraint), FNM returns partitions of similar quality to SC. To our knowledge, FNM is the only known algorithm that achieves different trade-offs between partition quality (i.e., *Ncut*) and fairness (i.e., *balance*) w.r.t. σ. We illustrate the performance of four algorithms as a function of the numbers of clusters



(b) Facebook (c) SBM (d) DBLP (e) LastFM (f) Credit

Figure 1. Quality (i.e., Ncut) and fairness (i.e., balance) of $k = 5$ partitions created by SC, FSC, sFSC, and FNM as a function of parameter σ. Note that when σ varies, FNM offers trade-offs between fairness and quality but SC, FSC, and sFSC remain unchanged in both measures.



(b) Facebook (c) SBM (d) DBLP (e) LastFM (f) Credit

Figure 2. Quality (i.e, Ncut) and fairness (i.e., balance) of SC, FSC, sFSC, and FNM ($\sigma = 0.8, 0.2$) as a function of the number of clusters k.

k in Figure 2. We vary k from 2 to 10 on three smaller datasets and from 5 to 50 on two larger datasets. For each algorithm, the Ncut value increases with k. Meanwhile, the balance of each algorithm generally drops with increasing k, and FNM is the only algorithm that consistently achieves a balance of at least $1 - \sigma$, i.e., guaranteeing the satisfaction of fairness constraints. Furthermore, FNM has comparable Ncut values to SC, FSC, and sFSC and higher balances on most datasets when $\sigma = 0.8$. But on the *LastFM* dataset, FNM has higher Ncut values than other algorithms when $\sigma = 0.8$ because it performs reassignments to ensure a balance of at least 0.2. When $\sigma = 0.2$, as FNM assigns more nodes to non-closest clusters for fairness than when $\sigma = 0.8$, it often has inferior partition quality, especially when k is large.

Note that the results that examine the quality of the partition by varying σ and k in the remaining four datasets, as well as the time efficiency by varying σ and k in all the nine datasets, are deferred to Appendix C of the full version of this paper [36].

Ablation Study. In the ablation study, we first run each embedding method to obtain a k-dimensional node embedding (for $k = 5, 20$) on each graph and use the same fair rounding in Algorithm 2 to produce two fair partitioning schemes with $\sigma = 0.8, 0.2$ from node vectors. SC, FSC, sFSC, and FNM are thus renamed SE, FSE, sFSE, and rFSE since we only tested their spectral embedding performance. We report the Ncut values on five small datasets (since FSE and FW cannot provide any result on four large datasets) in Table 3. Our method, rFSE in Algorithm 1, achieves the best or second-best partition quality among all methods in almost all cases. Here, SE and FSE/sFSE are special cases of rFSE when $\sigma = 1$ (i.e., w/o fairness) and 0 (i.e., with strict proportionality), respectively. As such, SE performs closely to rFSE when $\sigma = 0.8$, and FSE/sFSE provides similar embeddings to rFSE when $\sigma = 0.2$. Especially, sFSE slightly outperforms rFSE when $\sigma = 0.2$. Since sFSE and rFSE both provide embeddings with fractional fairness constraints, which are looser than the original integral ones, using a tighter (fractional) constraint than required (i.e., equal to $1 - \sigma$) in the embedding phase could help improve overall performance after rounding. The partition quality of deep learning-based node embeddings (with or without fairness) is much inferior to that of rFSE and other spectral embeddings since they are not designed for graph partitioning.

Then, we evaluate the performance of each rounding method on node vectors provided by rFSE on five small datasets for $k = 5, 20$ and $\sigma = 0.8, 0.2$ and present the Ncut values in Table 4. Our method, FR in Algorithm 2, performs best among the four fair methods we compare. We find that fair k-means for i.i.d. data in [7] may not be appropriate to round embedding vectors though it adopts the same fairness notion as ours. Despite having the lowest Ncut values, k-means++ cannot produce fair partitions. When $\sigma = 0.8$, FR provides partitions closer to k-means++ than when $\sigma = 0.2$ since fewer or no reassignments are required as fairness constraints are looser.

5 Conclusion

This paper investigated the $(\boldsymbol{\alpha}, \boldsymbol{\beta})$-proportionally fair normalized cut graph partitioning problem. We proposed a novel algorithm, FNM, consisting of an extended spectral embedding method and a k-means-based rounding scheme to provide a node partitioning with a small Ncut value on the given graph while strictly following the proportional fairness constraints. The comprehensive experimental findings confirmed the superior performance of FNM in terms of partition quality, fairness, and efficiency. In future work, we will generalize our algorithm to handle other notions of fairness, e.g., individual fairness [26, 39], in graph partitioning problems.

Table 3. Performance of different embedding methods on partition quality (i.e., *Ncut values*). Here, the best result(s) are highlighted in a **bold** font, and the second best result(s) are underlined.

Dataset	k	$\sigma = 0.8$							$\sigma = 0.2$						
		SE	**FSE**	**sFSE**	**DW**	**N2V**	**FW**	**rFSE (*)**	**SE**	**FSE**	**sFSE**	**DW**	**N2V**	**FW**	**rFSE (*)**
Facebook	5	1.378	1.401	1.401	**1.374**	1.438	1.444	1.378	1.676	**1.536**	**1.536**	1.676	1.575	1.689	1.546
	20	13.840	13.887	14.057	**13.050**	13.230	14.825	13.753	14.861	14.882	14.882	14.541	14.708	15.461	**14.346**
German	5	**1.433**	1.442	1.442	1.500	1.492	1.492	**1.433**	1.537	**1.471**	**1.471**	1.519	1.502	1.505	1.498
	20	11.856	11.879	11.869	11.977	11.922	12.638	**11.811**	12.927	12.889	12.884	12.954	13.012	13.059	**12.852**
SBM	5	**2.542**	2.619	2.619	2.585	2.807	2.941	**2.542**	3.377	**3.345**	**3.345**	3.490	3.509	3.515	3.348
	20	16.812	16.998	17.012	17.294	17.357	17.430	**16.785**	17.844	17.812	**17.784**	18.131	18.194	18.104	17.799
DBLP	5	0.050	**0.032**	**0.032**	0.929	1.002	0.423	0.050	1.003	**0.261**	**0.261**	0.995	1.029	0.645	0.269
	20	1.381	**0.941**	0.984	7.560	6.875	3.793	1.166	3.170	2.871	**2.779**	8.691	8.119	6.235	2.787
LastFM	5	0.294	0.453	0.453	0.846	0.754	1.392	**0.265**	0.908	**0.677**	**0.677**	1.704	1.667	1.900	0.699
	20	3.983	4.666	4.943	7.313	6.942	8.316	**2.922**	7.367	7.699	7.680	10.698	10.612	11.607	**7.080**
Avg. Ranking		2.2	2.8	3.1	4.1	4.4	5.5	**1.6**	4.0	2.2	**1.6**	4.8	4.8	5.4	1.7

Table 4. Performance of different rounding methods on partition quality (i.e., *Ncut values*). Here, the best result(s) are highlighted in a **bold** font, and the second best result(s) are underlined. Note that the results of k-means++ do not satisfy the fairness constraints and thus are just presented to show the "price of fairness" in the rounding process.

Dataset	k	$\sigma = 0.8$					$\sigma = 0.2$				
		k-means++	**K+R**	**FK**	**IP**	**FR (*)**	k-means++	**K+R**	**FK**	**IP**	**FR (*)**
Facebook	5	(1.378)	**1.378**	**1.378**	**1.378**	**1.378**	(1.381)	1.640	1.576	1.599	**1.546**
	20	(12.998)	14.161	13.947	**13.333**	13.753	(12.586)	14.991	14.887	14.615	**14.346**
German	5	(1.433)	**1.433**	1.442	**1.433**	**1.433**	(1.479)	1.526	**1.498**	1.499	**1.498**
	20	(11.629)	11.995	11.948	11.883	**11.811**	(11.733)	13.611	13.166	13.069	**12.852**
SBM	5	(2.542)	**2.542**	**2.542**	**2.542**	**2.542**	(2.568)	3.350	3.350	3.350	**3.348**
	20	(16.767)	16.839	16.834	16.803	**16.785**	(16.908)	18.006	17.926	17.889	**17.799**
DBLP	5	(0.022)	0.062	0.067	**0.049**	0.050	(0.024)	0.466	0.521	0.500	**0.269**
	20	(0.400)	1.381	1.066	**1.007**	1.166	(0.404)	4.596	4.104	3.452	**2.787**
LastFM	5	(0.152)	0.320	0.272	0.266	**0.265**	(0.184)	0.801	0.805	0.791	**0.699**
	20	(2.005)	5.751	5.139	3.211	**2.922**	(2.257)	10.723	11.144	9.911	**7.080**
Avg. Ranking		–	2.9	2.8	**1.4**	**1.4**	–	3.4	2.8	2.4	**1.0**

Acknowledgements

This work was supported by the National Natural Science Foundation of China under grant no. 62202169. We would like to thank the anonymous reviewers for their comments, which helped improve this work considerably.

Ethics Statement

The implication of this work is that it enables us to find proportionally representative graph partitions. This is relevant for various real-world applications to reduce harmful biases of traditional algorithms for this problem. As such, we do not foresee situations in which our method may be directly misused. This is based on the assumption that the user aims to improve social outcomes rather than having a negative impact.

References

[1] S. Ahmadian, A. Epasto, R. Kumar, and M. Mahdian, 'Fair correlation clustering', in *AISTATS*, pp. 4195–4205, (2020).

[2] S. Ahmadian and M. Negahbani, 'Improved approximation for fair correlation clustering', in *AISTATS*, pp. 9499–9516, (2023).

[3] A. Anagnostopoulos, L. Becchetti, A. Fazzone, C. Menghini, and C. Schwiegelshohn, 'Spectral relaxations and fair densest subgraphs', in *CIKM*, pp. 35–44, (2020).

[4] D. Arthur and S. Vassilvitskii, 'k-means++: the advantages of careful seeding', in *SODA*, pp. 1027–1035, (2007).

[5] A. Backurs, P. Indyk, K. Onak, B. Schieber, A. Vakilian, and T. Wagner, 'Scalable fair clustering', in *ICML*, pp. 405–413, (2019).

[6] J. Barzilai and J.M. Borwein, 'Two-point step size gradient methods', *IMA J. Numer. Anal.*, **8**(1), 141–148, (1988).

[7] S.K. Bera, D. Chakrabarty, N. Flores, and M. Negahbani, 'Fair algorithms for clustering', *Adv. Neural Inf. Process. Syst.*, **32**, 4955–4966, (2019).

[8] I.O. Bercea, M. Groß, S. Khuller, A. Kumar, C. Rösner, D.R. Schmidt, and M. Schmidt, 'On the cost of essentially fair clusterings', in *APPROX-RANDOM*, pp. 18:1–18:22, (2019).

[9] R. Berk, H. Heidari, S. Jabbari, M. Kearns, and A. Roth, 'Fairness in criminal justice risk assessments: The state of the art', *Sociol. Methods Res.*, **50**(1), 3–44, (2021).

[10] J. Cheeger, 'A lower bound for the smallest eigenvalue of the Laplacian', in *Problems in Analysis*, 195–200, Princeton University Press, (1971).

[11] I.Y. Chen, E. Pierson, S. Rose, S. Joshi, K. Ferryman, and M. Ghassemi, 'Ethical machine learning in healthcare', *Annu. Rev. Biomed. Data Sci.*, **4**, 123–144, (2021).

[12] X. Chen, B. Fain, L. Lyu, and K. Munagala, 'Proportionally fair clustering', in *ICML*, pp. 1032–1041, (2019).

[13] X. Chen, W. Hong, F. Nie, D. He, M. Yang, and J.Z. Huang, 'Spectral clustering of large-scale data by directly solving normalized cut', in *KDD*, pp. 1206–1215, (2018).

[14] S.E. Chew and N.D. Cahill, 'Semi-supervised normalized cuts for image segmentation', in *ICCV*, pp. 1716–1723, (2015).

[15] K.Y. Chiang, J.J. Whang, and I.S. Dhillon, 'Scalable clustering of signed networks using balance normalized cut', in *CIKM*, pp. 615–624, (2012).

[16] F. Chierichetti, R. Kumar, S. Lattanzi, and S. Vassilvitskii, 'Fair clustering through fairlets', *Adv. Neural Inf. Process. Syst.*, **30**, 5029–5037, (2017).

[17] E. Chlamtác, Y. Makarychev, and A. Vakilian, 'Approximating fair clustering with cascaded norm objectives', in *SODA*, pp. 2664–2683, (2022).

[18] A. Chouldechova and A. Roth, 'A snapshot of the frontiers of fairness in machine learning', *Commun. ACM*, **63**(5), 82–89, (2020).

[19] I. Davidson and S.S. Ravi, 'Making existing clusterings fairer: Algorithms, complexity results and insights', in *AAAI*, pp. 3733–3740, (2020).

[20] S.A. Esmaeili, B. Brubach, A. Srinivasan, and J. Dickerson, 'Fair clustering under a bounded cost', *Adv. Neural Inf. Process. Syst.*, **34**, 14345–14357, (2021).

[21] S.A. Esmaeili, B. Brubach, L. Tsepenekas, and J. Dickerson, 'Probabilistic fair clustering', *Adv. Neural Inf. Process. Syst.*, **33**, 12743–12755, (2020).

[22] S.A. Friedler, C. Scheidegger, S. Venkatasubramanian, S. Choudhary, E.P. Hamilton, and D. Roth, 'A comparative study of fairness-enhancing interventions in machine learning', in *FAT**, pp. 329–338, (2019).

[23] Z. Friggstad and R. Mousavi, 'Fair correlation clustering with global and local guarantees', in *WADS*, pp. 414–427, (2021).

[24] M. Ghadiri, S. Samadi, and S.S. Vempala, 'Socially fair k-means clustering', in *FAccT*, pp. 438–448, (2021).

[25] A. Grover and J. Leskovec, 'node2vec: Scalable feature learning for networks', in *KDD*, pp. 855–864, (2016).

[26] S. Gupta and A. Dukkipati, 'Consistency of constrained spectral clustering under graph induced fair planted partitions', *Adv. Neural Inf. Process. Syst.*, **35**, 13527–13540, (2022).

[27] S. Gupta, G. Ghalme, N.C. Krishnan, and S. Jain, 'Efficient algorithms for fair clustering with a new notion of fairness', *Data Min. Knowl. Disc.*, (2023).

[28] L.W. Hagen and A.B. Kahng, 'New spectral methods for ratio cut partitioning and clustering', *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.*, **11**(9), 1074–1085, (1992).

[29] J. Han, K. Xiong, and F. Nie, 'Orthogonal and nonnegative graph reconstruction for large scale clustering', in *IJCAI*, pp. 1809–1815, (2017).

[30] L. Huang, S.H. Jiang, and N.K. Vishnoi, 'Coresets for clustering with fairness constraints', *Adv. Neural Inf. Process. Syst.*, **32**, 7587–7598, (2019).

[31] H. Jia, S. Ding, M. Du, and Y. Xue, 'Approximate normalized cuts without eigen-decomposition', *Inf. Sci.*, **374**, 135–150, (2016).

[32] N. Karmarkar, 'A new polynomial-time algorithm for linear programming', *Combinatorica*, **4**(4), 373–396, (1984).

[33] M. Kleindessner, P. Awasthi, and J. Morgenstern, 'Fair k-center clustering for data summarization', in *ICML*, pp. 3448–3457, (2019).

[34] M. Kleindessner, S. Samadi, P. Awasthi, and J. Morgenstern, 'Guarantees for spectral clustering with fairness constraints', in *ICML*, pp. 3458–3467, (2019).

[35] J. Li, F. Nie, and X. Li, 'Directly solving the original ratiocut problem for effective data clustering', in *ICASSP*, pp. 2306–2310, (2018).

[36] J. Li, Y. Wang, and A. Merchant. Spectral normalized-cut graph partitioning with fairness constraints. http://arxiv.org/abs/2307.12065.

[37] S. Lloyd, 'Least squares quantization in PCM', *IEEE Trans. Inf. Theory*, **28**(2), 129–136, (1982).

[38] H. Lütkepohl, *Handbook of Matrices*, John Wiley & Sons, Inc., 1997.

[39] S. Mahabadi and A. Vakilian, 'Individual fairness for k-clustering', in *ICML*, pp. 6586–6596, (2020).

[40] Y. Makarychev and A. Vakilian, 'Approximation algorithms for socially fair clustering', in *COLT*, pp. 3246–3264, (2021).

[41] M. Mathioudakis, C. Castillo, G. Barnabò, and S. Celis, 'Affirmative action policies for top-k candidates selection: with an application to the design of policies for university admissions', in *SAC*, pp. 440–449, (2020).

[42] N. Mehrabi, F. Morstatter, N. Saxena, K. Lerman, and A. Galstyan, 'A survey on bias and fairness in machine learning', *ACM Comput. Surv.*, **54**(6), 115:1–115:35, (2021).

[43] M. Negahbani and D. Chakrabarty, 'Better algorithms for individually fair *k*-clustering', *Adv. Neural Inf. Process. Syst.*, **34**, 13340–13351, (2021).

[44] M.E.J. Newman, 'Finding community structure in networks using the eigenvectors of matrices', *Phys. Rev. E*, **74**(3), 036104, (2006).

[45] A.Y. Ng, M.I. Jordan, and Y. Weiss, 'On spectral clustering: Analysis and an algorithm', *Adv. Neural Inf. Process. Syst.*, **14**, 849–856, (2001).

[46] J. Nocedal and S.J. Wright, *Numerical Optimization*, Springer, New York, NY, USA, 1999.

[47] B. Perozzi, R. Al-Rfou, and S. Skiena, 'Deepwalk: Online learning of social representations', in *KDD*, pp. 701–710, (2014).

[48] T.A. Rahman, B. Surma, M. Backes, and Y. Zhang, 'Fairwalk: Towards fair graph embedding', in *IJCAI*, pp. 3289–3295, (2019).

[49] M. Schmidt, C. Schwiegelshohn, and C. Sohler, 'Fair coresets and streaming algorithms for fair k-means', in *WAOA*, pp. 232–251, (2019).

[50] J. Shi and J. Malik, 'Normalized cuts and image segmentation', *IEEE Trans. Pattern Anal. Mach. Intell.*, **22**(8), 888–905, (2000).

[51] D.B. Shmoys and É. Tardos, 'An approximation algorithm for the generalized assignment problem', *Math. Program.*, **62**, 461–474, (1993).

[52] S. Thejaswi, B. Ordozgoiti, and A. Gionis, 'Diversity-aware k-median: Clustering with fair center representation', in *ECML/PKDD (II)*, pp. 765–780, (2021).

[53] A. Vakilian and M. Yalçiner, 'Improved approximation algorithms for individually fair clustering', in *AISTATS*, pp. 8758–8779, (2022).

[54] H. Van Lierde, T.W.S. Chow, and G. Chen, 'Scalable spectral clustering for overlapping community detection in large-scale networks', *IEEE Trans. Knowl. Data Eng.*, **32**(4), 754–767, (2020).

[55] J. Wang, D. Lu, I. Davidson, and Z. Bai, 'Scalable spectral clustering with group fairness constraints', in *AISTATS*, pp. 6613–6629, (2023).

[56] E.W. Weisstein. Cayley transform. https://mathworld.wolfram.com/CayleyTransform.html.

[57] E.W. Weisstein. Stiefel manifold. https://mathworld.wolfram.com/StiefelManifold.html.

[58] Z. Wen and W. Yin, 'A feasible method for optimization with orthogonality constraints', *Math. Program.*, **142**(1-2), 397–434, (2013).

[59] S.X. Yu and J. Shi, 'Multiclass spectral clustering', in *ICCV*, pp. 313–319, (2003).

[60] Y. Zhang, C. Fang, Y. Wang, Z. Wang, Z. Lin, Y. Fu, and J. Yang, 'Multimodal style transfer via graph cuts', in *ICCV*, pp. 5942–5950, (2019).

[61] I.M. Ziko, J. Yuan, E. Granger, and I.B. Ayed, 'Variational fair clustering', in *AAAI*, pp. 11202–11209, (2021).