SCALABLE SPECTRAL CLUSTERING WITH GROUP FAIRNESS CONSTRAINTS

Ji Wang
Department of Coumputer Science
University of California, Davis
Davis, CA 95616
jiiwang@ucdavis.edu

Ding Lu
Department of Mathematics
University of Kentucky
Lexington, KY 40506
Ding.Lu@uky.edu

Zhaojun Bai
Department of Coumputer Science
University of California, Davis
Davis, CA 95616
zbai@ucdavis.edu

Ian Davidson
Department of Coumputer Science
University of California, Davis
Davis, CA 95616
indavidson@ucdavis.edu

November 1, 2022

ABSTRACT

There are synergies of research interests and industrial efforts in modeling fairness and correcting algorithmic bias in machine learning. In this paper, we present a scalable algorithm for spectral clustering (SC) with group fairness constraints. Group fairness is also known as statistical parity where in each cluster, each protected group is represented with the same proportion as in the entirety. While FairSC algorithm (Kleindessner et al., 2019) is able to find the fairer clustering, it is compromised by high costs due to the kernels of computing nullspaces and the square roots of dense matrices explicitly. We present a new formulation of underlying spectral computation by incorporating nullspace projection and Hotelling's deflation such that the resulting algorithm, called s-FairSC, only involves the sparse matrix-vector products and is able to fully exploit the sparsity of the fair SC model. The experimental results on the modified stochastic block model demonstrate that s-FairSC is comparable with FairSC in recovering fair clustering. Meanwhile, it is sped up by a factor of 12 for moderate model sizes. s-FairSC is further demonstrated to be scalable in the sense that the computational costs of s-FairSC only increase marginally compared to the SC without fairness constraints.

1 INTRODUCTION

Machine learning (ML) is widely used to automate decisions in areas such as the targeting of advertising, the issuing of credit cards, and the admission of students. While powerful, ML is vulnerable to biases encoded in the raw data or brought by the underlying algorithms against certain groups or individuals, thus resulting in *unfair* decisions (Hardt et al., 2016; Chouldechova and Roth, 2018). See Flores et al. (2016); Pethig and Kroenung (2022) for examples of algorithmic unfairness in real life. In the context of algorithmic decision-making, *fairness* commonly refers to the prohibition of any favoritism toward certain groups or individuals based on their natural or acquired characteristics. Such characteristics are also known as sensitive attributes, for instance, gender, ethnicity, sexual orientation, and age group (Mehrabi et al., 2021). The rising stake and growing societal impact of ML algorithms has motivated the study of fairness in ML in academia and industry. Various efforts have been attempted at modeling fairness and correcting algorithmic biases in both supervised and unsupervised ML (e.g., Dwork et al. (2012); Chierichetti et al. (2017); Samadi et al. (2018); Agarwal et al. (2019); Aghaei et al. (2019); Amini et al. (2019); Zhang et al. (2019); Davidson and Ravi (2020)).

There are a wide range of studies in fair ML depending on the choice of algorithms and fairness definitions. In this paper, we focus on spectral clustering (SC) with group fairness constraints. The notion of group fairness is an idea of stastical parity (Feldman et al., 2015; Zemel et al., 2013). It ensures that the overall proportion of members in a group receiving positive (negative) consideration is identical to the proportion of the population as a whole. In Kleindessner et al. (2019), a model is proposed to incorporate the group fairness notion into the SC framework, FairSC for short. For synthetic networks, FairSC is shown to recover ground-truth clustering with high probability. Additionally, FairSC identifies a fairer clustering compared to SC without fairness constraints on real-life data. Unfortunately, FairSC can only handle moderate problem sizes due to the computational costs in the computations of orthonormal bases of large nullspaces and the square roots of dense matrices. FairSC is not scalable.

In this paper, we present a new formulation of spectral computation of SC with fairness constraints by incorporating nullspace projection and Hotelling's deflation. The new algorithm is named Scalable FairSC, or s-FairSC. In s-FairSC, all computational kernels only involve the sparse matrix-vector multiplications, and therefore are capable of fully exploiting the sparsity of the fair SC model. A comparison of s-FairSC with FairSC on the modified stochastic block model exhibits 12x speed up for moderate model sizes. Meanwhile, s-FairSC is comparable with FairSC in recovering fair clustering. The s-FairSC is further demonstrated to be scalable in the sense that it only has a marginal increase of computational costs compared to the SC without fairness constraints.

The remainder of the paper is organized as follows. Section 2 reviews the basics of spectral clustering and group fairness. Section 3 first recaps FairSC and then derives s-FairSC. Section 4 starts with the descriptions of experiment datasets and then demonstrates the improvements of s-FairSC in computational efficiency and scalability while maintaining the same accuracy as FairSC.

2 SC AND FAIR SC

2.1 Clustering and fair clustering

Given a set of data, the goal of clustering is to partition the set into subsets such that data in the same subset are more similar to each other than in those of the other subsets. Mathematically, let $\mathcal{G}(V, W)$ denote a weighted and undirected graph with a set of vertices (data) $V = \{v_1, v_2, \ldots, v_n\}$ and a *weighted adjacency matrix* $W = (w_{ij}) \in \mathbb{R}^{n\times n}$. The matrix W encodes the edge information. We assume $w_{ij} \geq 0$ and $w_{ii} = 0$. If $w_{ij} > 0$, then (v_i, v_j) is an edge. We denote by $d_i = \sum_{j=1}^{n} w_{ij}$ the *degree* of a vertex v_i and $D = \operatorname{diag}(d_1, d_2, \ldots, d_n)$, the *degree matrix* of $\mathcal{G}(V, W)$. We assume there is no isolated vertex and consequently, D is positive definite.

The task of clustering is to partition V into k disjoint subsets (*clusters*):

$$V = C_1 \cup \cdots \cup C_k, \tag{1}$$

in such a way that the total weights within each subset are large and between two different subsets are small. The clustering (1) can be encoded in a *clustering indicator matrix* $H = (h_{i\ell}) \in \mathbb{R}^{n\times k}$:

$$h_{i\ell} := \begin{cases} 1, & \text{if } v_i \in C_\ell, \\ 0, & \text{otherwise,} \end{cases} \tag{2}$$

for $i = 1, 2, \ldots, n$ and $\ell = 1, 2, \ldots, k$.

Now let us consider how to enforce *group fairness* in clustering. The *groups* here refers to a partition of the collected data (e.g., based on sensitive attributes such as gender and race). We denote the groups with h non-empty subsets $V_1, \ldots, V_h$:

$$V = V_1 \cup \cdots \cup V_h, \tag{3}$$

where $V_i \cap V_j = \emptyset$ for $i \neq j$. The *group fairness* for clustering refers to the ideal case that objects from all groups are present proportionately in each cluster, also known as the statistical parity. Groups can also be encoded in a *group indicator matrix* $G = (g_{is}) \in \mathbb{R}^{n\times h}$:

$$g_{is} := \begin{cases} 1, & \text{if } v_i \in V_s, \\ 0, & \text{otherwise,} \end{cases} \tag{4}$$

for $i = 1, 2, \ldots, n$ and $s = 1, 2, \ldots, h$.

The following definition is due to Kleindessner et al. (2019), which extends the notion of group fairness by Chierichetti et al. (2017).

Definition 2.1. A clustering (1) is *group fair* with respect to a group partition (1) if in each cluster the objects from each group are present proportionately as in the original dataset. That is, for $\ell = 1, 2, \cdots, k$ and $s = 1, 2, \cdots, h$,

$$\frac{|V_s \cap C_\ell|}{|C_\ell|} = \frac{|V_s|}{|V|}, \tag{5}$$

where $|V|$ denotes the number of vertices in V.

The fairness condition (5) can be represented compactly using the indicator matrices H in (2) and G in (4). Let us introduce

$$M := G^T H \quad \text{and} \quad Z := (G^T \mathbf{1}_n) \cdot (H^T \mathbf{1}_n)^T. \tag{6}$$

Then the entries of M and Z are $m_{s\ell} = |V_s \cap C_\ell|$ and $z_{s\ell} = |V_s| \cdot |C_\ell|$ for $s = 1, 2, \ldots, h$ and $\ell = 1, 2, \ldots, k$, Consequently, the fairness condition (5) is equivalent to

$$n \cdot M = Z, \tag{7}$$

where $n = |V|$. By (6), we know that (7) holds if and only if

$$F_0^T H = 0, \tag{8}$$

where $F_0 := G - \mathbf{1}_n z^T \in \mathbb{R}^{n\times h}$ and $z := (G^T \mathbf{1}_n)/n \in \mathbb{R}^h$. Observe that the entries of the vector z satisfy $z_i = |V_i|/n$, for $i = 1, 2, \ldots, h$, according to the definition of G in (4).

The following lemma shows that it is sufficient to use the first $h-1$ columns of F_0 in the constraint (8). The idea of using the first $h-1$ columns of F_0 is from Kleindessner et al. (2019). Extended from this idea, we justify the choice of $h-1$ through the rank of F_0 and prove that $h-1$ is indeed the least number of columns necessary.

Lemma 2.1. *Let H be the clustering indicator matrix by* (2)*, and G be the group indicator matrix by* (4)*. Let $F_0 \in \mathbb{R}^{n\times h}$ be as defined in* (8) *and $F := F_0(:, 1 : h-1) \in \mathbb{R}^{n\times(h-1)}$ consists of the first $h-1$ columns*[1] *of F_0. Then,*

(i) $\operatorname{rank}(F_0) = \operatorname{rank}(F) = h-1$*;*

(ii) The clustering (1) *is group fair with respect to* (3) *if and only if*

$$F^T H = 0. \tag{9}$$

Proof. See Appendix A.1. □

2.2 SC and fair SC

The objective function of a normalized cut (NCut) is

$$\text{NCut}(C_1, \cdots, C_k) := \sum_{\ell=1}^{k} \frac{\text{Cut}(C_\ell, V\backslash C_\ell)}{\text{vol}(C_\ell)}, \tag{10}$$

where

$$\text{Cut}(C_\ell, V\backslash C_\ell) := \sum_{v_i\in C_\ell, v_j\in V\backslash C_\ell} w_{ij}, \tag{11}$$

$$\text{vol}(C_\ell) := \sum_{v_i\in C_\ell} d_i. \tag{12}$$

The NCut function calculates the scaled total weights of *between-cluster* edges. NCut measures the similarities between the clusters: a smaller NCut implies a better clustering. Hence, the goal is to minimize NCut. Note that the scaling in (10) by $\text{vol}(C_\ell)$ takes into account the size of the cluster to avoid outliers.

The NCut function (10) admits a nice expression using the clustering indicator matrix. Let us first scale the clustering indicator matrix H from (2) to

$$H \leftarrow H\widehat{D}^{-1}, \tag{13}$$

where $\widehat{D} = \operatorname{diag}\left(\sqrt{\text{vol}(C_1)}, \cdots, \sqrt{\text{vol}(C_k)}\right)$. We call the new H the *scaled indicator matrix*. For convenience, we use the same notation for both indicator matrices. Then, the NCut function is related to matrix trace in the following:

$$\text{NCut}(C_1, \cdots, C_k) = \operatorname{Tr}(H^T L H), \tag{14}$$

where $L = D - W$ is the *Laplacian* of $\mathcal{G}(V, W)$. If $\mathcal{G}(V, W)$ is connected, then L is semi-positive definite and has exactly one zero eigenvalue.

[1]By changing the order of groups $\{V_\ell\}$, the result also holds for F with arbitrary $(h-1)$ columns of F_0.

By (14), the NCut minimization is equivalent to the trace minimization problem

$$\min \operatorname{Tr}(H^T L H) \quad \text{s.t.} \quad H \text{ has form of (13).} \tag{15}$$

Solving problem (15) directly turns out to be NP-hard (Wagner and Wagner, 1993). The following relaxed version of problem (15) is usually solved instead:

$$\min_{H \in \mathbb{R}^{n \times k}} \operatorname{Tr}(H^T L H) \quad \text{s.t.} \quad H^T D H = I_k. \tag{16}$$

Once the optimal solution H above is obtained, the discrete solution for (15) can be obtained by approximation. Here, k-means algorithm (for the rows of H) is usually applied but other techniques are also available; see, e.g., Bach and Jordan (2003); Lang (2005).

Problem (16) is a classical trace minimization problem initially studied in Fan (1949). The following theorem can be found in (Horn and Johnson, 2012, p. 248).

Theorem 2.1. *For a symmetric matrix $M \in \mathbb{R}^{n \times n}$,*

$$\min_{X^T X = I_k} \operatorname{Tr}(X^T M X) = \operatorname{Tr}(X_*^T M X_*) = \sum_{i=1}^{k} \lambda_i,$$

where $\lambda_1 \le \cdots \le \lambda_k$ are the k smallest eigenvalues of M, and $X_* \in \mathbb{R}^{n \times k}$ contains the corresponding eigenvectors.

Back to the problem (16), we can first reformulate it to the standard trace minimization problem by a change of variables $X = D^{1/2} H$:

$$\min_{X \in \mathbb{R}^{n \times k}} \operatorname{Tr}(X^T L_{\mathrm{n}} X) \quad \text{s.t.} \quad X^T X = I_k, \tag{17}$$

where $L_{\mathrm{n}} = D^{-\frac{1}{2}} L D^{-\frac{1}{2}}$ and is known as the normalized Laplacian. According to Theorem 2.1, the optimal solution X of (17) consists of the eigenvectors of the k smallest eigenvalues of L_n. We can then recover the solution of the relaxed problem (16) by $H = D^{-1/2} X$ and use it to get a clustering.

We summarize the SC in Algorithm 1. The SC Shi and Malik (2000) and Ng et al. (2001) is a highly successful clustering algorithm. It is efficient and scalable because it can take advantage of the sparsity of the SC model and use the state-of-the-art scalable sparse eigensolvers.

Algorithm 1 SC (Spectral Clustering)

```
Input: weighted adjacency matrix $W \in \mathbb{R}^{n \times n}$; degree matrix $D \in \mathbb{R}^{n \times n}$; $k \in \mathbb{N}$
Output: a clustering of indices $1 : n$ into $k$ clusters
1: compute the Laplacian matrix $L = D - W$
2: compute the normalized Laplacian $L_{\mathrm{n}} = D^{-\frac{1}{2}} L D^{-\frac{1}{2}}$
3: compute the $k$ smallest eigenvalues of $L_{\mathrm{n}}$ and the corresponding eigenvectors $X \in \mathbb{R}^{n \times k}$
4: apply $k$-means clustering to the rows of $H = D^{-\frac{1}{2}} X$
```

The group fairness constraints can be elegantly incorporated into the SC. To do this, one can simply add the linear constraint (9) to the trace minimization problem (16), which leads to

$$\min_{H \in \mathbb{R}^{n \times k}} \operatorname{Tr}(H^T L H) \quad \text{s.t.} \quad H^T D H = I_k \text{ and } F^T H = 0, \tag{18}$$

where $L \in \mathbb{R}^{n \times n}$ is the graph Laplacian, $F \in \mathbb{R}^{n \times (h-1)}$ is from (9), and $F^T H = 0$ is the original (9) right-multiplied with $\widehat{D}^{-1}$ due to the scaling (13) of H.

The idea of enforcing group fairness in spectral cluster using the optimization (18) was proposed in Kleindessner et al. (2019). We will show that the additional linear constraints in (18) will introduce marginal extra cost than solving the SC only.

3 FAIR SC ALGORITHMS

In this section, we consider numerical algorithms for the constrained trace minimization problem (18). We first review the FairSC proposed in Kleindessner et al. (2019), and then address the algorithmic scalability issue of FairSC.

3.1 FairSC algorithm

A null-space based algorithm for solving problem (18) proposed in Kleindessner et al. (2019) is as follows. Since the columns of H live in the null space of F^T, we can write $H = ZY$ for some $Y \in \mathbb{R}^{(n-h+1)\times k}$, where $Z \in \mathbb{R}^{n\times(n-h+1)}$ is an orthonormal basis matrix of null(F^T). Consequently, the optimization problem (18) is equivalent to the following trace optimization without linear constraints:

$$\min_{Y\in\mathbb{R}^{(n-h+1)\times k}} \operatorname{Tr}(Y^T[Z^TLZ]Y) \quad \text{s.t.} \quad Y^T[Z^TDZ]Y = I_k. \tag{19}$$

We can turn (19) to the standard trace minimization (16) by a change of variables $X = QY$ with $Q = (Z^TDZ)^{1/2}$. This leads to

$$\min_{X\in\mathbb{R}^{(n-h+1)\times k}} \operatorname{Tr}(X^TMX) \quad \text{s.t.} \quad X^TX = I_k, \tag{20}$$

where $M = Q^{-1}Z^TLZQ^{-1}$. Observe that M is positive semi-definite of size $n-h+1$. According to Theorem 2.1, the solution of (20) are solved by linear eigenvalue problem $Mx = \lambda x$. The optimal solution $X = [x_1, \dots, x_k]$ consists of the eigenvectors of the k smallest eigenvalue of M. Finally, writing back to H, we have $H = ZQ^{-1}X$ is the solution of (18).

We summarize the algorithm for the group-fair spectral clustering in Algorithm 2, called FairSC. FairSC requires two major computational kernels. The first one is for the null space of F^T in step 2. This can be done by the SVD $F = U\Sigma V^T$, where $U \in \mathbb{R}^{n\times n}$ and $V \in \mathbb{R}^{(h-1)\times(h-1)}$ are orthogonal, and $\Sigma \in \mathbb{R}^{n\times(h-1)}$ is diagonal. According to Theorem 2.1, F has a full column rank $h-1$. So $U(:, h:n)$ is an orthonormal basis of the null space of F^T. We can also use QR decomposition $F = UR$, where $U \in \mathbb{R}^{n\times n}$ is orthogonal and $R \in \mathbb{R}^{n\times(h-1)}$ is upper triangular. We can then set $Z = U(:, h:n)$. For both SVD and QR, the computation complexity are about $\mathcal{O}(n(h-1)^2)$; see, e.g., Golub and Van Loan (1996). The second kernel is for the matrix square root of size $n-h+1$ in step 3. This can be done by the blocked Schur algorithm (Higham and Al-Mohy, 2010; Deadman et al., 2012). The computation complexity is $\mathcal{O}((n-h+1)^3)$. For matrices of large sizes, both kernels involving large dense matrices are computationally expensive due to memory space and data communication costs. Consequently, FairSC is only suitable for small to medium size fair SC models; see numerical results in Section 4.

Algorithm 2 FairSC

```
Input: weighted adjacency matrix $W \in \mathbb{R}^{n\times n}$; degree matrix $D \in \mathbb{R}^{n\times n}$; group-membership matrix $F \in \mathbb{R}^{n\times(h-1)}$;
    $k \in \mathbb{N}$
Output: a clustering of indices $1:n$ into $k$ clusters
 1: compute the Laplacian matrix $L = D - W$
 2: compute an orthonormal basis $Z$ of the null space of $F^T$
 3: compute the matrix square root $Q = (Z^TDZ)^{1/2}$
 4: compute $M = Q^{-1}Z^TLZQ^{-1}$
 5: compute the $k$ smallest eigenvalues of $M$ and the corresponding eigenvectors $X \in \mathbb{R}^{n\times k}$
 6: apply $k$-means clustering to the rows of $H = ZQ^{-1}X$
```

3.2 A simple variant of FairSC

As a simple variant of FairSC, we can first turn the optimization problem (18) to a standard trace minimization with linear constraints, and then remove the constraints using nullspace projection without computing the square root of a matrix. We begin by a change of variables $X = D^{\frac{1}{2}}H$ and turn the optimization (18) to

$$\min_{X\in\mathbb{R}^{n\times k}} \operatorname{Tr}(X^TL_{\mathrm{n}}X) \quad \text{s.t.} \quad X^TX = I_k \text{ and } C^TX = 0, \tag{21}$$

where $L_{\mathrm{n}} = D^{-\frac{1}{2}}LD^{-\frac{1}{2}}$ is the normalized Laplacian, and $C = D^{-\frac{1}{2}}F$. Recall that the degree matrix D is diagonal, so generating L_{n} and C requires only row and column scaling. Next, we remove the linear constraints $C^TX = 0$ in (21) using the null space basis. Since the columns of X live in the null space of C^T we can parameterize $X = VY$ for some $Y \in \mathbb{R}^{(n-h+1)\times k}$, where $V \in \mathbb{R}^{n\times(n-h+1)}$ is an orthonormal basis matrix of the null space of C^T. Then (21) is equivalent to the standard trace minimization

$$\min_{Y\in\mathbb{R}^{(n-h+1)\times k}} \operatorname{Tr}(Y^TL_{\mathrm{n}}^vY) \quad \text{s.t.} \quad Y^TY = I_k, \tag{22}$$

where $L_{\mathrm{n}}^v = V^TL_nV \in \mathbb{R}^{(n-h+1)\times(n-h+1)}$. Consequently, we can solve the symmetric eigenvalue problem

$$L_{\mathrm{n}}^vy = \lambda y. \tag{23}$$

The eigenvectors corresponding to the k smallest eigenvalues provide the solution Y of (22), by which we recover the solution $H = D^{-\frac{1}{2}}VY$ of the original minimization (18).

Although this simple variant of the FairSC avoids computing matrix square root, the other drawbacks of FairSC remain, namely explicit computation of the null space of C^T and eigenvalue computation of the dense matrix $L_{\rm n}^v$.

3.3 Scalable FairSC algorithm

We now show how to reformulate the eigenvalue problem (23) to address the remaining drawbacks of FairSC. We begin from the eigenvalue problem of $L_{\rm n}^v$ in (23):

$$(V^T L_{\rm n} V)\, y = \lambda y.$$

A left multiplication of V leads to

$$(VV^T L_{\rm n} VV^T)Vy = \lambda \cdot Vy, \tag{24}$$

where on the left side $VV^T \cdot Vy \equiv Vy$ due to the orthogonality of V. Denote by $P = VV^T$, a projection matrix onto the range space of V (i.e., null space of C^T). Then (24) leads to the following *projected eigenvalue problem*

$$L_{\rm n}^p\, x = \lambda x, \tag{25}$$

where $x = Vy$ and $L_{\rm n}^p = PL_{\rm n}P \in \mathbb{R}^{n\times n}$. Consequently, the eigenvalue λ of $L_{\rm n}^v$ in (23) is also an eigenvalue of $L_{\rm n}^p$ in (25).

One major advantage of the projected eigenvalue problem (25) is that it may avoid the computation of the null space of C^T by exploiting the fact that

$$P = I - U_2 U_2^T, \tag{26}$$

where $U_2 \in \mathbb{R}^{n\times(h-1)}$ is an orthonormal basis of the range C (recall $[V, U_2] \in \mathbb{R}^{n\times n}$ is orthogonal). This is especially beneficial when C is a tall skinny matrix, where $U_2 \in \mathbb{R}^{n\times(h-1)}$ is much smaller than $V \in \mathbb{R}^{n\times(n-h+1)}$. In addition, to compute the eigenvalues of $L_{\rm n}^p$ by an iterative eigensolver, we only need the matrix-vector product $L_n^p w$ for a given vector w, so $L_{\rm n}^p$ is never formed explicitly and Pw can be applied without formulating U_2; see implementation details in Section 3.3.

For FairSC, we need the k smallest eigenvalues of $L_{\rm n}^v$. To understand how those eigenvalues are corresponded to L_n^p, we have the following results to relate the eigenstructures of the two eigenproblems (23) and (25).

Proposition 3.1. *Suppose $L_{\rm n}^v$ has the eigendecomposition*

$$L_{\rm n}^v = Y\Lambda_v Y^T, \tag{27}$$

where $\Lambda_v = diag(\lambda_1, \lambda_2, \ldots, \lambda_{n-h+1})$ contains the eigenvalues, and Y is an orthogonal matrix of size $n-h+1$ containing eigenvectors. Then $L_{\rm n}^p$ from (25) has the eigendecomposition

$$L_{\rm n}^p = [U_1 \quad U_2] \begin{bmatrix} \Lambda_v & \\ & \mathbf{0}_{h-1,h-1} \end{bmatrix} \begin{bmatrix} U_1^T \\ U_2^T \end{bmatrix}, \tag{28}$$

where $U = [U_1, U_2] \in \mathbb{R}^{n\times n}$ is orthogonal with $U_1 = VY \in \mathbb{R}^{n\times(n-h+1)}$ and $U_2 \in \mathbb{R}^{n\times(h-1)}$ being an arbitrary orthonormal basis of the range of C.

Proof. See Appendix A.2. $\square$

The following is a direct consequence of Proposition 3.1.

Corollary 3.1. *Let $L_{\rm n}^v$ and $L_{\rm n}^p$ be as defined in (22) and (25). Then (i) If (λ, y) is an eigenpair of $L_{\rm n}^v$, then (λ, x) with $x = Vy$ is an eigenpair of $L_{\rm n}^p$. (ii) If (λ, x) is an eigenpair of $L_{\rm n}^p$ and $C^T x = 0$, then (λ, y) with $y = V^T x$ is an eigenpair of $L_{\rm n}^v$.*

Return to the computation of k smallest eigenvalues of $L_{\rm n}^v$. Since the original matrix $L_{\rm n}^v$ is positive semi-definite, it has $n-h+1$ order eigenvalues $0 \le \lambda_1 \le \cdots \le \lambda_{n-h+1}$. By (28), the projected matrix $L_{\rm n}^p$ has n ordered eigenvalues: $\underbrace{0 = \cdots = 0}_{h-1} \le \lambda_1 \le \cdots \le \lambda_{n-h+1}$, where the first $h-1$ zero eigenvalues (counting multiplicity) have eigenvectors in the range of C.

In the simple case of $\lambda_1 > 0$, the k smallest eigenvalues $\lambda_1, \ldots, \lambda_k$ of $L_{\rm n}^v$ corresponds to the k smallest *positive eigenvalues* of $L_{\rm n}^v$. To find those eigenvalues, we can first compute $K = k + h - 1$ smallest eigenvalues of $L_{\rm n}^p$ by an eigensolver, and then select the desired k eigenparis corresponding to non-zero eigenvalues (alternatively, select those eigenvalues with eigenvectors orthogonal to C). However, if $\lambda_1 = 0$ (or $\lambda_1 \approx 0$), then the simple selection scheme above does not work, as the eigenvector corresponding to $\lambda_1 = 0$ is mixed (or numerically mixed) with the eigenspace of the $h-1$ zero eigenvalues. The eigenspace mixing issue happens, in particular, if the solution are computed by an iterative method with a low accuracy.

In the following, we first discuss Hotelling's deflation (Hotelling, 1943), and then show how to apply it to address the eigenvalue selection issue for the projected eigenvalue problem (25). Hotelling's deflation is also known as explicit external deflation and is suitable for high-performance computing; see, e.g., Parlett (1998); Yamazaki et al. (2019). The main idea of Hotelling's deflation is summarized in the following proposition.

Proposition 3.2. *Let a symmetric matrix $M \in \mathbb{R}^{n\times n}$ has the eigenvalue decomposition*

$$M = Q\Lambda Q^T = [Q_1 \quad Q_2]\begin{bmatrix}\Lambda_1 & \\ & \Lambda_2\end{bmatrix}\begin{bmatrix}Q_1^T \\ Q_2^T\end{bmatrix}, \tag{29}$$

where $\Lambda_1 \in \mathbb{R}^{k\times k}$ and $\Lambda_2 \in \mathbb{R}^{(n-k)\times(n-k)}$ contain eigenvalues, and $Q_1 \in \mathbb{R}^{n\times k}$ and $Q_2 \in \mathbb{R}^{n\times(n-k)}$ are orthonormal eigenvectors. For a given shift $\sigma \in \mathbb{R}$, define the shifted matrix $M_\sigma = M + \sigma Q_1 Q_1^T$. Then M_σ has an eigenvalue decomposition

$$M_\sigma = [Q_1 \quad Q_2]\begin{bmatrix}\Lambda_1 + \sigma I & \\ & \Lambda_2\end{bmatrix}\begin{bmatrix}Q_1^T \\ Q_2^T\end{bmatrix}. \tag{30}$$

Proof. See Appendix A.3. □

Suppose we are interested in the eigenvalues Λ_2 and the corresponding eigenvectors Q_2. By Proposition 3.2 with the shift σ sufficiently large, eigenvalues in Λ_2 will always correspond to the $(n-k)$ smallest eigenvalues of M_σ, since the unwanted eigenvalues Λ_1 are shifted away to $\Lambda_1 + \sigma I$. All eigenvectors remain unchanged. This is exactly what we need to untangle the unwanted $(p-1)$ zero eigenvalues in (28) from the rest of the eigenvalues of λ_i.

Recall the eigenvalue decomposition of $L_{\rm n}^p$ in (28). To shift away the unwanted $p-1$ zero eigenvalues, we can apply Hotelling's deflation with a shift σ to obtain

$$L_{\rm n}^\sigma := L_{\rm n}^p + \sigma U_2 U_2^T, \tag{31}$$

where recall that U_2 is an orthonormal basis for the range of C. If the shift σ is chosen such that $\sigma > \lambda_k$, where λ_k is the k-th smallest eigenvalue in Λ_v, then our desired k smallest eigenvalues in Λ_v are corresponding to the k smallest eigenvalues of $L_{\rm n}^\sigma$. Consequently, the eigenvector selection issue is solved.

On the other hand, by (26), the shifted matrix in (31) can be expressed as follows

$$L_{\rm n}^\sigma = PL_{\rm n}P + \sigma(I-P) = P(L_{\rm n} - \sigma I)P + \sigma I. \tag{32}$$

Then the matrix-vector multiplication with $L_{\rm n}^\sigma$ only requires operations with P and $L_{\rm n}$. This is extremely beneficial for large-scale fair SC models.

Algorithm and implementation. As we have described, Hotelling's deflation with proper choices of σ resolves the eigenvalues selection issue for the projected eigenvalue problem (25). To summarize, the solution of the constrained trace minimization (18) can now be characterized in the following proposition.

Proposition 3.3. *Let $L_{\rm n}^\sigma \in \mathbb{R}^{n\times n}$ be as defined in* (32) *and assume σ is sufficiently large such that $\sigma > \lambda_k(L_{\rm n}^v)$, where $\lambda_k(L_{\rm n}^v)$ is the k-th smallest eigenvalue of $L_{\rm n}^v$ in* (23). *Then H is a solution to the trace minimization* (18) *if and only if $H = D^{-\frac{1}{2}}X$, where $X = [x_1, x_2, \ldots, x_k] \in \mathbb{R}^{n\times k}$ contains the k eigenvectors corresponding to the k smallest eigenvalues of $L_{\rm n}^\sigma$.*

Our final algorithm based on projected eigenproblem (25) and Hotelling's deflation is presented in Algorithm 3,called scalable FairSC, s-FairSC in short.

Implementation issues. We discuss a few implementation issues regarding s-FairSC. (i) For the eigenvalue computation of $L_{\rm n}^\sigma$, we can use an iterative eigensolver, such as `eigs` in MATLAB, which is based ARPACK (Lehoucq et al., 1998), an implicitly restarted Arnoldi method. The eigensolver only needs to access $L_{\rm n}^\sigma$ through the matrix-vector multiplication $L_{\rm n}^\sigma w$ with a vector w. By (32), we have

$$\begin{aligned} L_{\rm n}^\sigma w &= P(L_{\rm n} - \sigma I)Pw + \sigma w \\ &= P(L_{\rm n}(Pw)) - Pw + \sigma w. \end{aligned}$$

Algorithm 3 Scalable FairSC (s-FairSC)

```
Input: weighted adjacency matrix $W \in \mathbb{R}^{n\times n}$; degree matrix $D \in \mathbb{R}^{n\times n}$; group-membership matrix $F \in \mathbb{R}^{n\times(h-1)}$;
    shift $\sigma \in \mathbb{R}$; $k \in \mathbb{N}$
Output: a vector of length $n$: a clustering of $n$ vertices into $k$ clusters
 1: compute the Laplacian matrix $L = D - W$
 2: set $L_{\rm n} = D^{-\frac{1}{2}} L D^{-\frac{1}{2}}$, and $C = D^{-\frac{1}{2}} F$
 3: compute the $k$ smallest eigenvalues of $L_{\rm n}^{\sigma}$ in (32) and the corresponding eigenvectors as columns of $X \in \mathbb{R}^{n\times k}$
 4: apply $k$-means clustering to the rows of $H = D^{-\frac{1}{2}} X$
```

It requires to call two times the projection operation with P. (ii) The projection P in (26) can be written as $P = I - C(C^TC)^{-1}C^T$, see, e.g., Golub et al. (2000). Consequently, Pw only requires operations with the matrix C:

$$Pw = (I - C(C^TC)^{-1}C^T)w = w - Cz, \tag{33}$$

where $z = (C^TC)^{-1}C^Tw$ is the solution to the least-squares problem $\min_z \|Cz - w\|_2$. For small to moderate size problems, direct LS solver can be applied for z. For large scale problems, iterative methods such as LSQR (Paige and Saunders, 1982) can be applied; this is in line with the inner-outer iterations for computing the eigenvalues of $PL_{\rm n}P$ (Golub et al., 2000). (iii) For an appropriate choice of shift σ, one can use an estimation for the largest eigenvalue of $L_{\rm n}$ as recommended in Lin et al. (2021). Such shift also guarantees the numerical stability of Hotelling's deflation (Lin et al., 2021).

3.4 Related work

The idea of transforming an optimization problem (18) to an equivalent eigenvalue problem is very natural. For the case of $k = 1$, the projected eigenvalue problem (25) was considered in Golub (1973) and (Golub and Van Loan, 1996, p. 621).

The projected eigenvalue problem from (25) is a form of so-called *constrained eigenvalue problems*, which is more generally formulated as $Ax = \lambda Mx$ subject to $C^Tx = 0$, where A and M are symmetric and M is positive definite. The constrained eigenvalue problems are found in many applications. There are a number of approaches available; see, e.g., Arbenz and Drmac (2002) for an algorithm for positive semidefinite A with a known null space; Baker and Lehoucq (2009) for a preconditioning technique; Golub et al. (2000) for a Lanczos process with inner-outer iterations to handle large matrices; and Porcelli et al. (2015) for a solution procedure within the structural finite-element code NOSA-ITACA.

For many constrained eigenvalue problems, the matrix C is corresponding to the null space of A, and the constraint $C^Tx = 0$ is to avoid computing "null eigenvectors". Since the entire nullspace of A is avoided, there is no eigenvector selection issue as in our problem (25). Particularly, Simoncini (2003) proposed a reformulation of the constrained eigenproblem based on null eigenvalue shifting. Her approach essentially includes Hotelling's deflation as a special case, but with a goal to shift away the entire null-space of A. By discussion in Section 3, we show that Hotelling's deflation is also capable of splitting the unwanted null vectors from those desired ones.

4 EXPERIMENTS

In this section, we present experimental results on the proposed s-FairSC algorithm. s-FairSC is implemented in MATLAB. [2] The results are obtained from a MacBook Pro with 2.3GHz 8-core i9 processor, 16 GB memory and an L3 cache of 16 MB.

4.1 Datasets

Modified Stochastic block model. The stochastic block model (SBM) (Holland et al., 1983) is a random graph model with planted blocks (ground-truth clustering). It is widely used to generate synthetic networks for clustering and community detection (Rohe et al., 2011; Balakrishnan et al., 2011; Lei and Rinaldo, 2015; Sarkar and Bickel, 2015). To take group fairness into account, we utilize a modified SBM (m-SBM) proposed by Kleindessner et al. (2019). In m-SBM, n vertices are assigned to k clusters for a prescribed fair ground-truth clustering $V = C_1 \cup \cdots \cup C_k$, and edges are placed between vertex pairs with probabilities dependent only on whether (or not) they belong to the same

[2] Code is available on `https://github.com/jiiwang/scalable_fair_spectral_clustering`.

cluster or group, hence graph $\mathcal{G}(V, W)$ is generated (see Appendix B.1 for details). Suppose $V = \widehat{C}_1 \cup \cdots \cup \widehat{C}_k$ is a computed clustering. The quality with respect to the fair ground-truth is measured by the *error rate*:

$$e(\widehat{H} - H) := \frac{1}{n} \min_{J \in \Pi_k} \|\widehat{H}J - H\|_F^2, \tag{34}$$

where $H, \widehat{H}$ are the indicator matrices for the fair ground-truth clustering and computed clustering respectively, and Π_k is the set of all $k \times k$ permutation matrices. The error rate counts the proportion of misclustered vertices.

FacebookNet. FacebookNet[3] is a small dataset that corresponds to Facebook friendship relations between students in a high school in France in 2013. The dataset was collected by Mastrandrea et al. (2015) to study human social networks and channels of information propagation, and opinion formation. It has also been used in clustering (Crawford and Milenković, 2018; Kleindessner et al., 2019; Chodrow et al., 2021). The dataset consists of two parts. One part contains relations of students in graph $\mathcal{G}(V, W)$. V is the set of students with $n = |V| = 155$, and an edge encoded in W represents a friendship between two students. Another part records the gender of the students $V = V_1 \cup V_2$, where V_1 and V_2 are for girls and boys, respectively. $|V_1| = 70$ and $|V_2| = 85$.

Random Laplacian. To create a random graph $\mathcal{G}(V, W)$, we generate a random symmetric weight matrix $W \in \mathbb{R}^{n\times n}$ with a prescribed sparsity $s = 10\%$, where $|V| = n$. Based on W, degree matrix D, and Laplacian matrix L are computed. The matrix $F \in \mathbb{R}^{n\times(h-1)}$ is also constructed as a random matrix. Note that in this dataset, F does not contain group-membership information, but only preserves the dimension. The dataset will only be used for experiments on scalability.

4.2 Experimental results

Experiment 4.1. This experiment is conducted on the modified SBM. We compare the error rate in (34) and running time of SC, FairSC [4] and s-FairSC.

Figure 1 depicts the error rate and running timing of SC, FairSC and s-FairSC. SC and s-FairSC are tested from model sizes $n = 1000$ to 10000. FairsSC stops at $n = 4000$ due to its high computational cost, echoing results reported in Kleindessner et al. (2019).

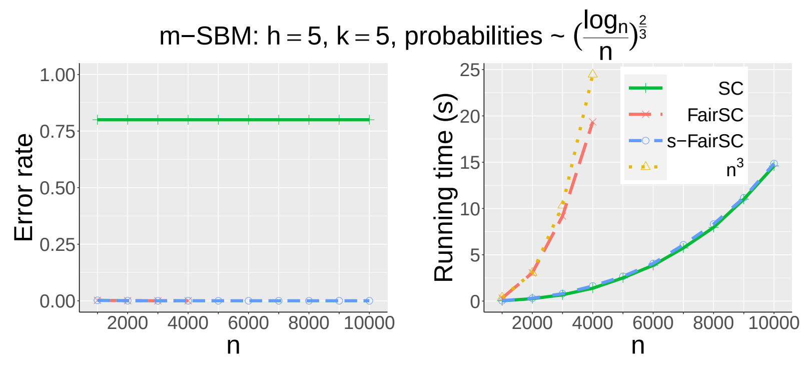

Figure 1: Error rate and running time of SC, FairSC and s-FairSC of an m-SBM with $h = 5$, $k = 5$, and edge connectivity probabilities proportional to $(\frac{\log n}{n})^{\frac{2}{3}}$.

From Figure 1, we observe that SC fails to recover the fair ground-truth clustering, while both FairSC and s-FairSC are able to retrieve the fair ground-truth clustering. In terms of the fairness quality of the computed clustering, s-FairSC is as fair as FairSC. However, the running time of s-FairSC is only a fraction of that of FairSC. For instance, when $n = 4000$, s-FairSC is 12x faster than FairSC. We also observe that s-FairSC is as scalable as SC, namely without fairness constraints.

Experiment 4.2. In this experiment, we use the FacebookNet dataset to quantify the approximation to the exact group fairness. Table 1 records the quantities of Definition 2.1.

[3] `http://www.sociopatterns.org/datasets/high-school-contact-and-friendship-networks/`.

[4] SC and FairSC code is available on `https://github.com/matthklein/fair_spectral_clustering`.

	SC	FairSC	s-FairSC
$\frac{\lvert V_1\rvert}{\lvert V\rvert}$	0.4516		
$\frac{\lvert V_1 \cap \widehat{C_1}\rvert}{\lvert\widehat{C_1}\rvert}$	0.6528	0.3537	0.3537
$\frac{\lvert V_1 \cap \widehat{C_2}\rvert}{\lvert\widehat{C_2}\rvert}$	0.2771	0.5616	0.5616
$\frac{\lvert V_2\rvert}{\lvert V\rvert}$	0.5484		
$\frac{\lvert V_2 \cap \widehat{C_1}\rvert}{\lvert\widehat{C_1}\rvert}$	0.3472	0.6463	0.6463
$\frac{\lvert V_2 \cap \widehat{C_2}\rvert}{\lvert\widehat{C_2}\rvert}$	0.7229	0.4384	0.4384

Table 1: Fractions of group membership within each cluster for the recovered clustering $V = \widehat{C_1} \cup \widehat{C_2}$.

Alternatively, the concept of the average balance introduced in Chierichetti et al. (2017) can be used as a metric to assess the fairness of a clustering. Given a computed clustering $V = \widehat{C_1} \cup \cdots \cup \widehat{C_k}$ and group partition $V = V_1 \cup \cdots \cup V_h$, the balance of cluster $\widehat{C_\ell}$ for $\ell = 1, 2, \cdots, k$ is defined as

$$\text{balance}(\widehat{C_l}) := \min_{s \neq s' \in \{1,\cdots,h\}} \frac{|V_s \cap \widehat{C_l}|}{|V_{s'} \cap \widehat{C_l}|} \in [0, 1] \tag{35}$$

The average balance is $\frac{1}{k}\sum_{l=1}^{k} \text{balance}(\widehat{C_l})$. A higher balance implies a fairer clustering; see Appendix B.2 for explanation.

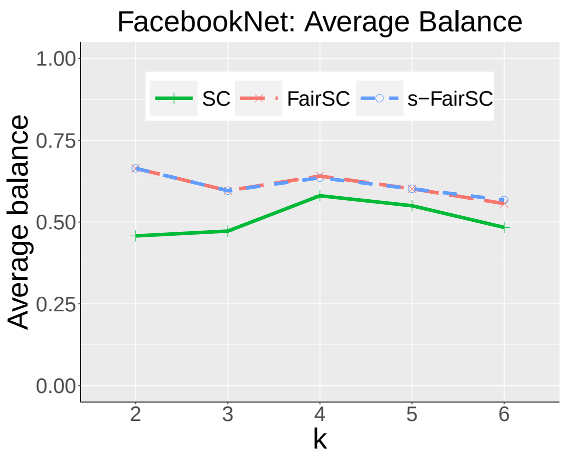

Figure 2: Average balance of SC, FairSC and s-FairSC as a function of k.

By Table 1 and Figure 2, we observe that since problem (15) is relaxed to problem (16), the equality in (5) is no longer true. Nevertheless, both FairSC and s-FairSC have improved fairness compared to SC. In addition, FairSC and s-FairSC produce almost identical results.

Experiment 4.3. In this experiment, we use random Laplacian to demonstrate that s-FairSC is as scalable as SC. Figure 3 reports the running time of SC and s-FairSC for problem size n from 5000 to 10000. We observe that s-FairSC is only slightly more expensive than SC.

5 CONCLUDING REMARKS

FairSC (Kleindessner et al., 2019) is able to recover fairer clustering, but sacrifices performance and scalability. In this paper, we presented a scalable FairSC (s-FairSC). s-FairSC incorporates nullspace projection and Hotelling's deflation such that all computational kernels only involve the sparse matrix-vector products and are able to fully exploit the sparsity of the fair SC model. We used the modified SBM to demonstrate that s-FairSC is as accurate as FairSC in

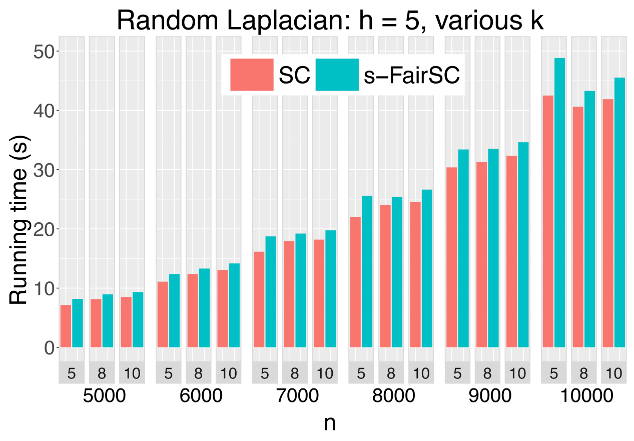

Figure 3: Running time of SC and s-FairSC on random Laplacian with $h = 5$ and $k \in \{5, 8, 10\}$.

retrieving fair clustering. Meanwhile, s-FairSC is sped up by a factor of 12 for the moderate SC model sizes, and demonstrated to be scalable in the sense that it only has a marginal increase of computational costs compared to SC without fairness constraints.

An intriguing problem for the future work is on the development of scalable algorithms to solve the the group fairness condition 5 in a less stringent manner. For instance, Bera et al. (2019) introduced a notion of group fairness for clustering with lower and upper bounds:

$$\beta_s \leq \frac{|V_s \cap C_\ell|}{|C_\ell|} \leq \alpha_s \quad \text{for } s = 1, 2, \cdots, h, \tag{36}$$

where β_s and α_s are lower and upper bounds for group V_s, respectively, such that $0 < \beta_s \leq \alpha_s < 1$. (See numerical data reported in Table 1). Another research topic is to extend the study of s-FairSC to individual fairness, where any two individuals who are similar with respect to a specific sensitive attribute should be treated similarly (Dwork et al., 2012; Zemel et al., 2013). Gupta and Dukkipati (2021) devised a SC model with individual fairness constraints. However, the existing algorithm fails to scale up due to the high costs of computational kernels.

References

A. Agarwal, M. Dudík, and Z. S. Wu. Fair regression: Quantitative definitions and reduction-based algorithms. In *International Conference on Machine Learning*, pages 120–129. PMLR, 2019.

S. Aghaei, M. J. Azizi, and P. Vayanos. Learning optimal and fair decision trees for non-discriminative decision-making. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence and Thirty-First Innovative Applications of Artificial Intelligence Conference and Ninth AAAI Symposium on Educational Advances in Artificial Intelligence*, pages 1418–1426, 2019.

A. Amini, A. P. Soleimany, W. Schwarting, S. N. Bhatia, and D. Rus. Uncovering and mitigating algorithmic bias through learned latent structure. In *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society*, pages 289–295, 2019.

P. Arbenz and Z. Drmac. On positive semidefinite matrices with known null space. *SIAM Journal on Matrix Analysis and Applications*, 24(1):132–149, 2002. doi: 10.1137/S0895479800381331. URL `https://doi.org/10.1137/S0895479800381331`.

F. Bach and M. Jordan. Learning spectral clustering. *Advances in neural information processing systems*, 16, 2003.

C. Baker and R. Lehoucq. Preconditioning constrained eigenvalue problems. *Linear algebra and its applications*, 431 (3-4):396–408, 2009.

S. Balakrishnan, M. Xu, A. Krishnamurthy, and A. Singh. Noise thresholds for spectral clustering. *Advances in Neural Information Processing Systems*, 24, 2011.

S. Bera, D. Chakrabarty, N. Flores, and M. Negahbani. Fair algorithms for clustering. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL `https://proceedings.neurips.cc/paper/2019/file/fc192b0c0d270dbf41870a63a8c76c2f-Paper.pdf`.

F. Chierichetti, R. Kumar, S. Lattanzi, and S. Vassilvitskii. Fair clustering through fairlets. *Advances in Neural Information Processing Systems*, 30, 2017.

P. S. Chodrow, N. Veldt, and A. R. Benson. Generative hypergraph clustering: From blockmodels to modularity. *Science Advances*, 7(28):eabh1303, 2021.

A. Chouldechova and A. Roth. The frontiers of fairness in machine learning. *arXiv preprint arXiv:1810.08810*, 2018.

J. Crawford and T. Milenković. Cluenet: Clustering a temporal network based on topological similarity rather than denseness. *PloS one*, 13(5):e0195993, 2018.

I. Davidson and S. Ravi. Making existing clusterings fairer: Algorithms, complexity results and insights. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 3733–3740, 2020.

E. Deadman, N. J. Higham, and R. Ralha. Blocked schur algorithms for computing the matrix square root. In *International Workshop on Applied Parallel Computing*, pages 171–182. Springer, 2012.

C. Dwork, M. Hardt, T. Pitassi, O. Reingold, and R. Zemel. Fairness through awareness. In *Proceedings of the 3rd innovations in theoretical computer science conference*, pages 214–226, 2012.

K. Fan. On a theorem of weyl concerning eigenvalues of linear transformations i. *Proceedings of the National Academy of Sciences of the United States of America*, 35(11):652, 1949.

M. Feldman, S. A. Friedler, J. Moeller, C. Scheidegger, and S. Venkatasubramanian. Certifying and removing disparate impact. In *proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, pages 259–268, 2015.

A. W. Flores, K. Bechtel, and C. T. Lowenkamp. False positives, false negatives, and false analyses: A rejoinder to machine bias: There's software used across the country to predict future criminals. and it's biased against blacks. *Fed. Probation*, 80:38, 2016.

G. H. Golub. Some modified matrix eigenvalue problems. *SIAM Review*, 15(2):318–334, 1973.

G. H. Golub and C. F. Van Loan. *Matrix computations*. Johns Hopkins University Press, 1996.

G. H. Golub, Z. Zhang, and H. Zha. Large sparse symmetric eigenvalue problems with homogeneous linear constraints: the Lanczos process with inner–outer iterations. *Linear Algebra And Its Applications*, 309(1-3):289–306, 2000.

S. Gupta and A. Dukkipati. Protecting individual interests across clusters: Spectral clustering with guarantees. *arXiv preprint arXiv:2105.03714*, 2021.

M. Hardt, E. Price, and N. Srebro. Equality of opportunity in supervised learning. *Advances in neural information processing systems*, 29, 2016.

N. J. Higham and A. H. Al-Mohy. Computing matrix functions. *Acta Numerica*, 19:159–208, 2010.

P. W. Holland, K. B. Laskey, and S. Leinhardt. Stochastic blockmodels: First steps. *Social Networks*, 5(2):109–137, 1983. ISSN 0378-8733. doi: https://doi.org/10.1016/0378-8733(83)90021-7. URL `https://www.sciencedirect.com/science/article/pii/0378873383900217`.

R. A. Horn and C. R. Johnson. *Matrix analysis*. Cambridge university press, 2012.

H. Hotelling. Some new methods in matrix calculation. *The Annals of Mathematical Statistics*, 14(1):1–34, 1943.

M. Kleindessner, S. Samadi, P. Awasthi, and J. Morgenstern. Guarantees for spectral clustering with fairness constraints. In *International Conference on Machine Learning*, pages 3458–3467. PMLR, 2019.

K. Lang. Fixing two weaknesses of the spectral method. *Advances in Neural Information Processing Systems*, 18, 2005.

R. B. Lehoucq, D. C. Sorensen, and C. Yang. *ARPACK users' guide: solution of large-scale eigenvalue problems with implicitly restarted Arnoldi methods*. SIAM, 1998.

J. Lei and A. Rinaldo. Consistency of spectral clustering in stochastic block models. *The Annals of Statistics*, 43(1): 215–237, 2015.

C.-P. Lin, D. Lu, and Z. Bai. Backward stability of explicit external deflation for the symmetric eigenvalue problem. *arXiv preprint arXiv:2105.01298*, 2021.

R. Mastrandrea, J. Fournet, and A. Barrat. Contact patterns in a high school: A comparison between data collected using wearable sensors, contact diaries and friendship surveys. *PLOS ONE*, 10:1–26, 09 2015. doi: 10.1371/journal.pone.0136497. URL `https://doi.org/10.1371/journal.pone.0136497`.

N. Mehrabi, F. Morstatter, N. Saxena, K. Lerman, and A. Galstyan. A survey on bias and fairness in machine learning. *ACM Computing Surveys (CSUR)*, 54(6):1–35, 2021.

A. Ng, M. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. *Advances in neural information processing systems*, 14, 2001.

C. C. Paige and M. A. Saunders. Lsqr: An algorithm for sparse linear equations and sparse least squares. *ACM Transactions on Mathematical Software (TOMS)*, 8(1):43–71, 1982.

B. N. Parlett. *The symmetric eigenvalue problem*. SIAM, 1998.

F. Pethig and J. Kroenung. Biased humans,(un) biased algorithms? *Journal of Business Ethics*, pages 1–16, 2022.

M. Porcelli, V. Binante, M. Girardi, C. Padovani, and G. Pasquinelli. A solution procedure for constrained eigenvalue problems and its application within the structural finite-element code nosa-itaca. *Calcolo*, 52(2):167–186, 2015.

K. Rohe, S. Chatterjee, and B. Yu. Spectral clustering and the high-dimensional stochastic blockmodel. *The Annals of Statistics*, 39(4):1878–1915, 2011.

S. Samadi, U. Tantipongpipat, J. H. Morgenstern, M. Singh, and S. Vempala. The price of fair pca: One extra dimension. *Advances in neural information processing systems*, 31, 2018.

P. Sarkar and P. J. Bickel. Role of normalization in spectral clustering for stochastic blockmodels. *The Annals of Statistics*, 43(3):962–990, 2015.

J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on pattern analysis and machine intelligence*, 22(8):888–905, 2000.

V. Simoncini. Algebraic formulations for the solution of the nullspace-free eigenvalue problem using the inexact shift-and-invert lanczos method. *Numerical linear algebra with applications*, 10(4):357–375, 2003.

D. Wagner and F. Wagner. Between min cut and graph bisection. In *International Symposium on Mathematical Foundations of Computer Science*, pages 744–750. Springer, 1993.

I. Yamazaki, Z. Bai, D. Lu, and J. Dongarra. Matrix powers kernels for thick-restart lanczos with explicit external deflation. In *2019 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, pages 472–481. IEEE, 2019.

R. Zemel, Y. Wu, K. Swersky, T. Pitassi, and C. Dwork. Learning fair representations. In *International conference on machine learning*, pages 325–333. PMLR, 2013.

H. Zhang, S. Basu, and I. Davidson. A framework for deep constrained clustering-algorithms and advances. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 57–72. Springer, 2019.

A MISSING PROOFS

A.1 Proof of Lemma 2.1

For (i): Recall that each row of G contains exactly one nonzero entry equals 1. Hence,

$$G\mathbf{1}_h = \mathbf{1}_n \quad \text{and} \quad \mathbf{1}_n^T G \mathbf{1}_h = n. \tag{37}$$

Because G has orthogonal columns, the first equation above also implies

$$\mathbf{1}_h = G^\dagger \mathbf{1}_n, \tag{38}$$

where $G^\dagger := (G^T G)^{-1} G^T$ denotes the pseudo inverse of G. For $\operatorname{rank}(F_0) = h - 1$, it is sufficient to show that the null space of F_0 is of dimension one. Let $x \in \mathbb{R}^p$ be a null vector of F_0, i.e., $F_0 x = 0$. By the definition of F_0, we have

$$Gx = \alpha \cdot \mathbf{1}_n \quad \text{with} \quad \alpha := (\mathbf{1}_n^T G x)/n.$$

A multiplication of $G^\dagger$ to the equation, together with (38), leads to $x = \alpha \mathbf{1}_h$. On the other hand, $\mathbf{1}_h$ is indeed a null vector of F_0:

$$F_0 \cdot \mathbf{1}_h = G\mathbf{1}_h - \mathbf{1}_n(\mathbf{1}_n^T G \mathbf{1}_h)/n = 0,$$

where we used (37). Consequently, the null space of F_0 is spanned by the vector $\mathbf{1}_h$. By the rank-nullity theorem, $\operatorname{rank}(F_0) = h - 1$. It follows from $F_0 \cdot \mathbf{1}_h = 0$ that the last column of F_0 is a linear combination of the first $h - 1$ columns. So we have $\operatorname{rank}(F_0) = \operatorname{rank}(F)$.

For (ii): It follows from (i) that F_0 and F have the same range space. Hence, $F_0^T y = 0$ if and only if $F^T y = 0$. Then (ii) follows from (8).

A.2 Proof of Proposition 3.1

We verify that (28) is an eigenvalue decomposition. First, since V is a basis of the null space of C^T and U_2 is a basis of the range of C, we have $V^T U_2 = 0$ and

$$U_1^T U_2 = Y^T (V^T U_2) = 0.$$

A quick verification shows $U = [U_1, U_2]$ satisfy $U^T U = I_n$, so U is orthogonal.

On the other hand, it follows from $L_\text{n}^v = V^T L_\text{n} V$ and $L_\text{n}^p = P L_\text{n} P$ that

$$L_\text{n}^p = V L_\text{n}^v V^T.$$

Hence,

$$L_\text{n}^p U_1 = (V L_\text{n}^v V^T)(VY) = V(L_\text{n}^v Y) = V(Y\Lambda_v) = U_1 \Lambda_v,$$

where we used (27) in the third equality. On the other hand, $V^T U_2 = 0$ also implies

$$L_\text{n}^p U_2 = V L_\text{n}^v V^T U_2 = 0.$$

Consequently, $L_\text{n}^p [U_1, U_2] = [U_1, U_2] \cdot \text{blockdiag}(\Lambda_v, \mathbf{0}_{h-1,h-1})$, i.e., (28).

A.3 Proof of Proposition 3.2

It follows from (29) that $MQ_1 = Q_1 \Lambda_1$ and $MQ_2 = Q_2 \Lambda_2$. Consequently, $M_\sigma Q_1 = MQ_1 + \sigma Q_1 = Q_1(\Lambda_1 + \sigma I)$ and $M_\sigma Q_2 = MQ_2 = Q_2 \Lambda_2$. We prove (30).

B DISCUSSIONS ON THE EXPERIMENTS

B.1 m-SBM Dataset

In this section, we first describe the traditional stochastic block model (SBM). Then, we discuss a necessary modification to accommodate group fairness.

SBM. In SBM, n vertices are assigned to k blocks (*clusters*) for a prescribed clustering, and edges are placed between vertex pairs with probabilities dependent only on the cluster membership of the vertices.

Specifically, to generate the desired SBM using a random graph $\mathcal{G}(V, W)$ for recovering a ground-truth clustering $V = C_1 \cup \cdots \cup C_k$, we need a pair of parameters (b, P), where $b \in \mathbb{N}^k$ is the block vector where each element b_i denotes the number of vertices in C_i such that $\sum_{i=1}^{k} b_i = n$, where $n = |V|$. P is a symmetric probability matrix $P = (p_{ij}) \in \mathbb{R}^{n\times n}$ to define the edge connectivity as follows

$$p_{ij} = \begin{cases} a, & v_i \text{ and } v_j \text{ in the same cluster,} \\ b, & v_i \text{ and } v_j \text{ in different clusters.} \end{cases} \tag{39}$$

Once b is assigned, we can represent the ground-truth clustering by an indicator matrix $H \in \{0,1\}^{n\times k}$ defined in (2). For the probability matrix P, we require $a > b$ so that two vertices within the same cluster have a higher chance to be joined by an edge than vertex pairs between clusters.

Given (b, P), the weighted adjacency matrix $W = (w_{ij}) \in \{0, \alpha, \beta\}^{n\times n}$ of graph G where α is the weight of within-cluster edges and β is the weight of between-cluster edges ($\alpha > \beta$) is given by

$$w_{ij} = \begin{cases} \text{Bernoulli}(p_{ij}), & i < j, \\ 0, & i = j, \\ w_{ji}, & i > j \end{cases} \tag{40}$$

where Bernoulli(p_{ij}) is the Bernoulli distribution of w_{ij} with probability p_{ij} such that

$$\begin{cases} P_r(w_{ij} = \alpha) = p_{ij} = 1 - P_r(w_{ij} = 0), v_i \text{ and } v_j \text{ in the same cluster,} \\ P_r(w_{ij} = \beta) = p_{ij} = 1 - P_r(w_{ij} = 0), v_i \text{ and } v_j \text{ in different clusters.} \end{cases} \tag{41}$$

Thus, we obtain SBM $\mathcal{G}(V, W)$.

Example B.1. Given $k = 3, n = 6$, we set block vector $b = [b_1, b_2, b_3] = [2, 2, 2]$, parameters $\alpha = 3, \beta = 1$ for the edge weight, and parameters $a = 0.6, b = 0.2$ for the probability matrix P. First, we define a ground-truth clustering from b using the cluster indicator matrix H as follows.

$$H = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix}^T .$$

Based on H, we then follow (39), (40) and (41) to generate the adjacency matrix W with given α, β, and a, b. Figure 4 depicts $\mathcal{G}(V, W)$ and the underlying prescribed ground-truth clustering $V = C_1 \cup C_2 \cup C_3$. $\square$

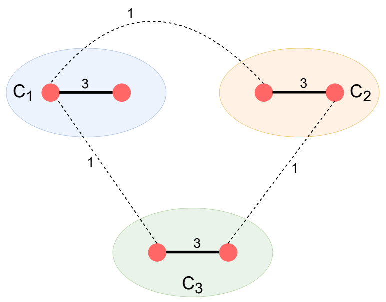

Figure 4: Example of graph $\mathcal{G}(V, W)$ generated by the traditional SBM. $V = C_1 \cup C_2 \cup C_3$ is a ground-truth clustering.

Modified SBM (m-SBM). The SBM described above cannot be used directly for the fair SC since it does not take group fairness into consideration. The prescribed ground-truth clustering built from the block vector b is not guaranteed to be fair since b is not encoded with any group information. Furthermore, the edge connectivity matrix P does not consider within-group and between-group probabilities.

To take into the account of group information for the fair SC, let us define $b = [b_1, b_2, \ldots, b_k]$ as the block vector consisting of k blocks. Each block $b_i \in \mathbb{N}^h$ contains h elements: $|b_i| = k$, and each element $b_i^{(j)}$ of b_i equals the number of vertices in $V_j \cap C_i$: $b_i^{(j)} = |V_j \cap C_i|$. Consequently, we have

$$\sum_{i=1}^{k} b_i^{(j)} = |V_j|, \quad \sum_{j=1}^{h} b_i^{(j)} = |C_i|, \quad \text{and} \quad \sum_{i=1}^{k}\sum_{j=1}^{h} b_i^{(j)} = n.$$

To satisfy the fairness condition (5), for $j = 1, 2, \cdots, h$, the elements of the vector b should be chosen to satisfy

$$\frac{b_i^{(j)}}{\sum_{j=1}^{h} b_i^{(j)}} = \frac{\sum_{i=1}^{k} b_i^{(j)}}{|V|} \quad \text{for any } i = 1, 2, \cdots, k. \tag{42}$$

Once b is set, we obtain

- Group-membership vectors $g^{(s)}$ for $s = 1, 2, \cdots, h$ defined in (4), and the corresponding group-membership matrix F defined in Lemma 2.1;
- Clustering indicator matrix $H \in \{0, 1\}^{n \times k}$ defined in (2). H contains a fair ground-truth clustering.

Next we define the probability matrix $P = (p_{ij}) \in \mathbb{R}^{n \times n}$ for edge connectivity

$$p_{ij} = \begin{cases} a, & v_i, v_j \text{ in the same cluster and group,} \\ b, & v_i, v_j \text{ in different clusters but the same group,} \\ c, & v_i, v_j \text{ in the same cluster but different groups,} \\ d, & v_i, v_j \text{ in different clusters and groups,} \end{cases} \tag{43}$$

where $a > b > c > d$.

With P, we can generate an adjacency matrix $W = (w_{ij}) \in \{0, \alpha, \beta\}^{n \times n}$ where α is the weight of within-cluster edges and β is the weight of between-cluster edges ($\alpha > \beta$) as follows

$$w_{ij} = \begin{cases} \text{Bernoulli}(p_{ij}), & i < j, \\ 0, & i = j, \\ w_{ji}, & i > j \end{cases} \tag{44}$$

where Bernoulli(p_{ij}) is the Bernoulli distribution of w_{ij} with probability p_{ij} such that

$$\begin{cases} P_r(w_{ij} = \alpha) = p_{ij} = 1 - P_r(w_{ij} = 0), v_i \text{ and } v_j \text{ in the same cluster,} \\ P_r(w_{ij} = \beta) = p_{ij} = 1 - P_r(w_{ij} = 0), v_i \text{ and } v_j \text{ in different clusters.} \end{cases} \tag{45}$$

In this way, we acquire m-SBM $\mathcal{G}(V, W)$.

Example B.2. Given $k = 3, h = 2, n = 10$, we set block vector $b = [b_1, b_2, b_3] = [(2, 2), (2, 2), (1, 1)]$, parameters $\alpha = 3, \beta = 1$ for the edge weight, and parameters $a = 0.6, b = 0.4, c = 0.2, d = 0.1$ for the probability matrix P.

By the vector b, we have the following group membership vectors and a fair ground-truth clustering as $g^{(s)}$ and H, respectively.

$$g^{(1)} = [1 \quad 1 \quad 0 \quad 0 \quad 1 \quad 1 \quad 0 \quad 0 \quad 1 \quad 0]^T,$$
$$g^{(2)} = [0 \quad 0 \quad 1 \quad 1 \quad 0 \quad 0 \quad 1 \quad 1 \quad 0 \quad 1]^T,$$
$$H = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix}^T.$$

Following (43), (44) and (45) with given weights α, β, probabilities a, b, c, d, group information $g^{(1)}, g^{(2)}$, and the cluster information in H, we can generate the adjacency matrix W. The m-SBM $\mathcal{G}(V, W)$ is shown in Figure 5. □

B.2 Fairness Measured by Balance

For the metric of balance discussed in section 4.2, we stated "a higher balance implies a fairer clustering". Here we give a brief justification.

First let us recall the definition of balance:

Given a clustering $V = C_1 \cup \cdots \cup C_k$ and group partition $V = V_1 \cup \cdots \cup V_h$, the balance of cluster C_ℓ for $\ell = 1, 2, \cdots, k$ is defined as

$$\text{balance}(C_l) = \min_{s \neq s' \in \{1, \cdots, h\}} \frac{|V_s \cap C_l|}{|V_{s'} \cap C_l|} \in [0, 1] \tag{46}$$

The average balance is

$$\frac{1}{k} \sum_{l=1}^{k} \text{balance}(C_l).$$

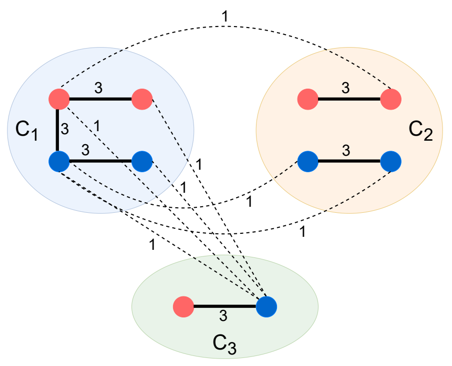

Figure 5: Example of graph $\mathcal{G}(V, W)$ generated by the m-SBM. $V = C_1 \cup C_2 \cup C_3$ is a fair ground-truth clustering w.r.t. $V = V_1 \cup V_2$ where V_1 is the set of red vertices while V_2 is the set of blue vertices.

It follows from Kleindessner et al. (2019) that the balance has an upper bound such that

$$\min_{\ell \in \{1,\cdots,k\}} \text{balance}(C_l) \leq \min_{s \neq s' \in \{1,\cdots,h\}} \frac{|V_s|}{|V_{s'}|} \tag{47}$$

By the group fairness defined in (5), we have that for any $\ell = 1, 2, \cdots, k$ and any $s = 1, 2, \cdots, h$,

$$\frac{|V_s \cap C_\ell|}{|C_\ell|} = \frac{|V_s|}{|V|} \equiv \frac{|V_s \cap C_l|}{|V_{s'} \cap C_l|} = \frac{|V_s|}{|V_{s'}|} \tag{48}$$

Combining (46), (47), and (48), we can conclude that a higher balance indicates a fairer clustering.

C ADDITIONAL EXPERIMENTS

In Experiment 4.1 (m-SBM experiment), we observed that

(i) in terms of the quality of the clustering, s-FairSC is as fair as FairSC.

(ii) the running time of s-FairSC is only a fraction of that of FairSC. (s-FairSC has about 12x speedup when $n = 4000$)

(iii) s-FairSC is as scalable as SC without fairness constraints.

In the following, we provide additional experiments to support the above findings with respect to the number of groups h.

Recall that the number of groups h is associated with the number of linear constraints in the fair SC model (18). We perform experiments on the m-SBM dataset with fixed probabilities a, b, c, d and number of clusters k, but different h. Similar to Experiment 4.1, SC and s-FairSC are tested from model sizes $n = 1000$ to 10000 while FairsSC stops at $n = 4000$ because of its high cost. From Figure 6, we confirm that s-FairSC finds a clustering as fair as FairSC. Moreover, s-FairSC runs significantly faster than FairSC when h varies; when $n = 4000$, it has a speedup of 12x, 12x, and 13x when $h = 2, 5$, and 10 respectively.

Noticeably, we observe that when $h = 10$, s-FairSC is even faster than SC without constraints. For instance, when $n = 10000$, s-FairSC takes only 62% time of SC. Some factors might play an role, such as the sparsity distribution of the random graph $\mathcal{G}(V, W)$, and the eigenvalue distribution of L_{n}^{σ} in (32). This issue is under investigation and remains an open question.

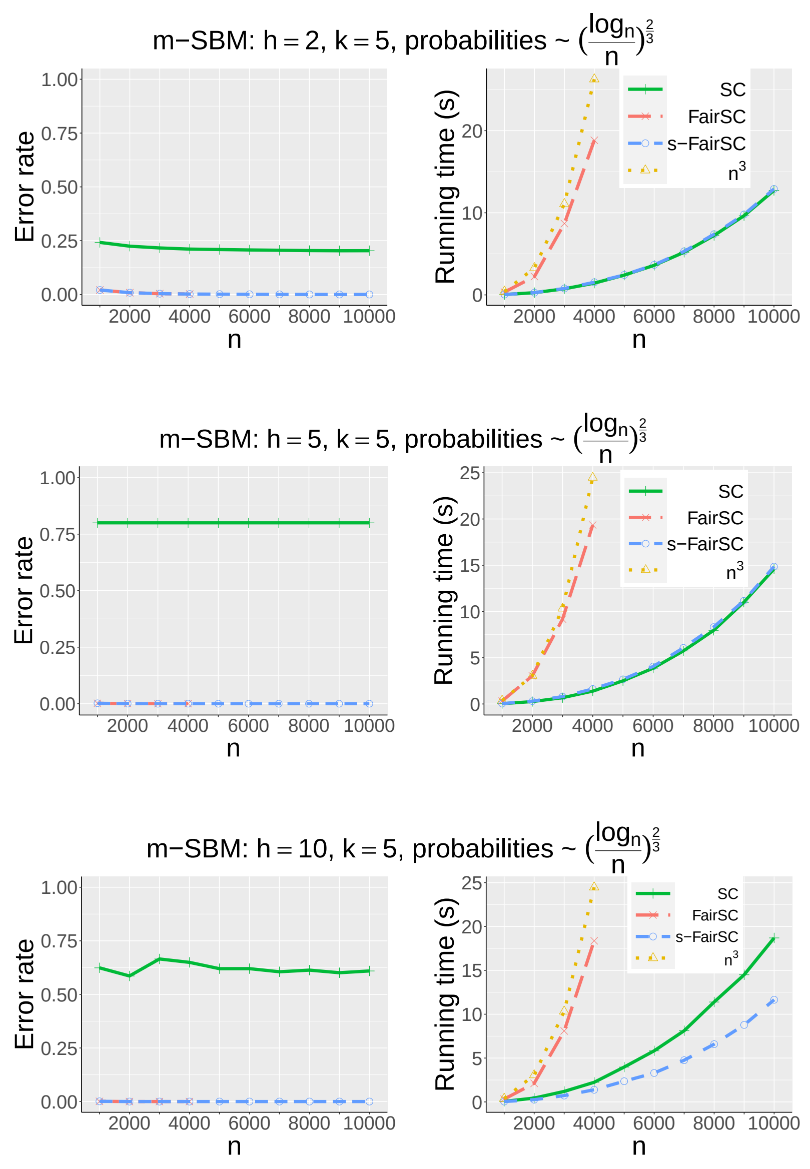

Figure 6: Error rate and running time of SC, FairSC, and s-FairSC on m-SBM with $h \in \{2, 5, 10\}$, edge connectivity probabilities proportional to $(\frac{\log n}{n})^{\frac{2}{3}}$, specifically $a = 10 \times (\frac{\log n}{n})^{2/3}, b = 7 \times (\frac{\log n}{n})^{2/3}, c = 4 \times (\frac{\log n}{n})^{2/3}, d = (\frac{\log n}{n})^{2/3}$. Note that the plots in the second row are the same as those in Figure 1.