Efficient First-Order Methods for Peng-Wei Semi-definite Programming with application to optimal clustering

Xinyu Li

May 2020

1 Abstract

The problem of clustering n points in $\mathbb{R}^d$ into k groups has become popular with the rise of unsupervised machine learning. To efficiently solve the clustering problem, I first convert the optimal clustering problem into a graph cut problem. I continue to consider the convex relaxation of the graph cut problem, which is known as Peng-Wei Semidefinite programming (SDP) relaxation. Though Peng-Wei SDP has been proved to have good statistical properties in previous literatures, it is a highly-constrained convex optimization problem with $O(n^2)$ constraints. In this project, I apply James Renegar's efficient first-order method to Peng-Wei SDP. The computational complexity of solving the Peng-Wei SDP is reduced from $O(n^7 \log(\frac{1}{\epsilon}))$ with a standard SDP solver to $O(\frac{n^4\sqrt{\log(n)}}{\epsilon})$ with Renegar's first order method.

2 Introduction

In unsupervised machine learning problems, given n feature vectors $\{x_i \in \mathbb{R}^d, i \in [n]\}$, one major task is to partition these n points into several meaningful groups, such that among each group the objects share similar features. Some common strategies to solve the clustering problems includes k-means which is a NP-hard problem and computational hard to solve, and the Lloyd Algorithm which gives a heuristic result. Spectral clustering nowadays has received more attention, which usually involves three steps: 1. Construct a weight matrix with corresponding graph Laplacian that convert the points into a graph cut problem; 2. Solve for the optimal cut with respect to the graph Laplacian; 3. Round the solution to a feasible solution for clustering. Ling and Stohmer provided theoretical foundations for the global optimality of graph cuts via Peng-Wei SDP [3]. However, the Peng-Wei SDP itself though convex, remains difficult to solve.

Following Ling and Stohmer's convex relaxation into the Peng-Wei SDP, I mainly focus on designing an algorithm to solve the Peng-Wei SDP efficiently. I apply Renegar's first order methods to solve the Peng-Wei SDP. Eisenach and Liu proposed a FORCE algorithm based on one of Renegar's first order methods [2] with computation complexity of $O(\frac{n^6}{k^2\epsilon})$. However, the complexity could be improved by applying Renegar's first order method in a different manner. Some computational steps such as projection could also be theoretically finished. The diameter of the variable spaces is also reduced by considering the special structure of matrices feasible for Peng-Wei SDP. The ultimate computation complexity with Renegar's First Order Smoothed Scheme is $O(\frac{n^4\sqrt{\log(n)}}{\epsilon})$.

The paper is organized as follows. In Section 3, I review the basics for converting the clustering problem into a graph cut problem and relaxing the NP-hard optimization problem into the convex Peng-Wei SDP. In Section 4, I convert the Peng-Wei SDP into forms that could be resolved by Renegar's first order methods. In Section 5, I present the necessary steps for applying Renegar's methods. Two algorithms are based on Renegar's Non-Smoothed Scheme, which internally runs SubGradient method. Other algorithm is based on Renegar's Smoothed Scheme. Finally, in Section 6, I prove the computational complexity for the algorithms solving the Peng-Wei SDP.

2.1 Notation

$\langle A, B\rangle$ represents the inner product of matrix A, B: $\langle A, B\rangle = \text{Trace}(A^T B)$. For each $m \in \mathbb{N}$, $[m] := \{1, 2, ..., m\}$. $\mathcal{A}(X) = \mathbf{b}$ represent $\{\langle A_i, X\rangle = b_i \text{ for } i \in [m]\}$. $\mathbb{1}_n$ represents a vector in $\mathbb{R}^n$ which all entries are 1. $|\Gamma|$ is the cardinality of set Γ. Clustering of the data points are denoted as $\Gamma = \{\Gamma_1, ..., \Gamma_k\}$, where each Γ_i is a cluster of points. $\lambda(\cdot)$ is a function returns the eigenvalues of the matrix. $\lambda_{\min}(A)$ denotes the smallest eigenvalue of matrix A. $\mathbb{S}^{n\times n}$ is the symmetric matrix of dimension $n \times n$. $\mathbb{S}_+^{n\times n}$ is the positive semi-definite matrix of dimension $n \times n$. $\text{vec}(A)$ represent vectorize a $d \times d$ matrix A into a vector in $\mathbb{R}^{d^2}$, where $A_{i,j} = \text{vec}(A)_{(i-1)d+j}$. $\text{diag}(v)$ turns a vector $v \in \mathbb{R}^d$ into a diagonal matrix in $\mathbb{R}^{d\times d}$ whose diagonal equals to v. $\text{dvec}(A) := \text{diag}(\text{vec}(A))$.

3 Problem Formulation

First, given n data points $\{x_1, x_2, ..., x_n\}$ in $\mathbb{R}^d$, we would first formulate the data clustering problem into a convex optimization problem. The ultimate goal is to find out a partition $\Gamma^* = \{\Gamma_1^*, ..., \Gamma_k^*\}$ such that inside each Γ_i^*, the data points are close to each other while for data points from distinct partitions, their difference is more significant.

Converting Data Clustering problem into Graph Cuts problem Given a disjoint partition $\{\Gamma_l\}_{l=1}^k$ of n points in $\mathbb{R}^d$, we first denote the similarity matrix between cluster Γ_a and cluster Γ_b as $W^{(a,b)}$ and

$$W_{i,j}^{(a,b)} := \phi(-\frac{||x_i^{(a)} - x_j^{(b)}||}{\sigma}), \quad W^{(a,b)} \in \mathbb{R}^{|\Gamma_a|\times|\Gamma_b|}, \tag{1}$$

where $x_i^{(a)}$ represents the position of the ith point in Γ_a, ϕ is a function that should satisfy the following properties: when x_i is close to x_j, the weights $W_{i,j} := \phi(-\frac{||x_i - x_j||}{\sigma})$ is large; while x_i is distant from x_j, the weights should be small. The common choice of ϕ includes:

- $\phi(t) = 1_{|t|\leq 1}$ for σ-neighbourhood graph;
- $\phi(t) = e^{-\frac{t^2}{2}}$ for weighted complete graph.

The full weight matrix W combines similarity matrices of all pairs of clusters:

$$W := \begin{pmatrix} W^{(1,1)} & W^{(1,2)} & ... & W^{(1,k)} \\ W^{(2,1)} & W^{(2,2)} & ... & W^{(2,k)} \\ \vdots & \vdots & \ddots & \vdots \\ W^{(k,1)} & W^{(k,2)} & ... & W^{(k,k)} \end{pmatrix}. \tag{2}$$

We denote $w_{i,j}$ as the (i, j) entry of the weight matrix W. The degree of vertex i is $d_i = \sum_{j=1}^N w_{i,j}$ and the degree matrix D is a diagonal matrix:

$$D := \text{diag}(W\mathbb{1}_n). \tag{3}$$

The unnormalized graph Laplacian for the weight matrix W is

$$L := D - W. \tag{4}$$

Then, we define ratio cuts [3] of the given partition as

$$\text{RatioCut}(\{\Gamma_l\}_{l=1}^k) := \sum_{l=1}^k \frac{\text{cut}(\Gamma_l, \Gamma_l^c)}{|\Gamma_l|}, \tag{5}$$

where

$$\begin{aligned}\text{cut}(\Gamma_l, \Gamma_l^c) &:= \sum_{i\in\Gamma_l, j\in\Gamma_l^c} w_{i,j} = \sum_{a\neq l} \mathbb{1}_{|\Gamma_a|}^T W^{(a,l)} \mathbb{1}_{|\Gamma_l|} \\ &= \sum_{a=1}^{k} \mathbb{1}_{|\Gamma_a|}^T W^{(a,l)} \mathbb{1}_{|\Gamma_l|} - \mathbb{1}_{|\Gamma_l|}^T W^{(l,l)} \mathbb{1}_{|\Gamma_l|} \\ &= \langle D^{(l,l)} - W^{(l,l)}, \mathbb{1}_{|\Gamma_l|} \mathbb{1}_{|\Gamma_l|}^T \rangle \\ &= \langle L^{(l,l)}, \mathbb{1}_{|\Gamma_l|} \mathbb{1}_{|\Gamma_l|}^T \rangle \end{aligned} \tag{6}$$

Thus,

$$\text{RatioCut}(\{\Gamma_l\}_{l=1}^k) = \sum_{l=1}^{k} \frac{1}{|\Gamma_l|} < L^{(l,l)}, \mathbb{1}_{|\Gamma_l|} \mathbb{1}_{|\Gamma_l|}^T >=< L, X_{\text{cut}} >, \tag{7}$$

where

$$X_{\text{cut}} = \text{blockdiag}(\frac{1}{|\Gamma_1|} \mathbb{1}_{|\Gamma_1|} \mathbb{1}_{|\Gamma_1|}^T, ..., \frac{1}{|\Gamma_k|} \mathbb{1}_{|\Gamma_k|} \mathbb{1}_{|\Gamma_k|}^T). \tag{8}$$

RatioCut is indeed the inner product between the graph Laplacian L and a block-diagonal matrix X_{cut}.

In order to maximize the inter-cluster connectivity while minimize the cross-cluster connectivity, our clustering problem of finding the partition could be converted into minimizing the RatioCut in equation 7 by finding a solution X_{cut} in form of equation 8. However, this turns out to be an NP-hard problem. Nevertheless, though it is hard to find out X_{cut} directly, it is indeed contained in a convex set thus we can first find an approximation of X_{cut}, later round it into the form in equation 7. We followed the relaxation of RatioCut proposed by Ling et al. in [3]: finding a matrix X satisfying the following properties

- $X \succeq 0$;
- $X \geq 0$;
- $X\mathbb{1}_n = \mathbb{1}_n$;
- $\text{Tr}(X) = k$.

We then consider the following optimization problem, the solutions of which contains the optimal X_{cut} :

$$\begin{aligned} \underset{X\in\mathbb{S}_n}{\text{minimize}} \ & < L, X > \\ \text{subject to } & X \succeq 0 \\ & X \geq 0 \\ & X\mathbb{1}_n = \mathbb{1}_n \\ & \text{Tr}(X) = k \end{aligned} \tag{9}$$

This convex optimization program is known as Peng-Wei SDP. A solution X is called "integer" if $X_{i,j} = \frac{1}{\Gamma_a}$ if $i, j \in \Gamma_a$ and 0 otherwise. A integer solution X directly yields a corresponding partition $\{\Gamma_a\}_{a=1}^k$.

4 SPD Conversion

To efficiently solve the Peng-Wei SDP, we apply James Renegar's first-order method in [5]. Renegar's first-order method solves SDP in the following form:

$$\begin{aligned} \underset{X\in\mathbb{S}_n}{\text{minimize}} \ & \langle L, X \rangle \\ \text{subject to } & \mathcal{A}(X) = \mathbf{b} \\ & X \succeq 0. \end{aligned} \tag{10}$$

One could see that Peng-Wei SDP (9) is not in the standard form of the SDP proposed by Renegar with an elementwise constraint $X \geq 0$. In this section, we present the conversion of Peng-Wei SDP into other forms, which are necessary for applying Renegar's first order method in later section.

Conversion to an Eigenvalue Maximization Problem One of the fundamental idea in Renegar's first order method is to convert the SDP with linear constraints into an eigenvalue optimization problem. Theorem 2.2 in Renegar[5] shows that SDP in (10) is equivalent to a particular eigenvalue optimization problem given the identity matrix I is a strictly feasible solution. However, in Peng-Wei SDP, I is not strictly feasible since it does not satisfy $I > 0$ elementwisely. Let's present a modified theorem here:

Theorem 1. *Let F be an input strictly feasible solution to (10), $\lambda_{min,F}(X) := \lambda_{min}(F^{-1/2}XF^{-1/2})$. Let* val *be any value satisfying* val $< \langle L, F\rangle$. *If Y^* solves*

$$\begin{aligned} &\underset{Y\in\mathbb{S}_n}{\textit{maximize}}\ \lambda_{min,F}(Y) \\ &\textit{subject to}\ \mathcal{A}(Y) = \mathbf{b} \\ &\qquad\qquad \langle L, Y\rangle = \text{val} \end{aligned} \tag{11}$$

then $P_F(Y^*) := F + \frac{1}{1-\lambda_{min,F}(Y^*)}(Y^* - F)$ is optimal for (10). Conversely, if X^* is optimal for SDP (10), then $Y^* := F + \frac{\langle L,F\rangle - \text{val}}{\langle L,F\rangle - \langle L,X^*\rangle}(X^* - F)$ is optimal for (11) and $X^* = P_F(Y^*)$.

Proof. Fix val $< \langle L, F\rangle$. Consider the feasible space for SDP (11):

$$\mathcal{C}_Y = \{Y \in \mathbb{S}^{n\times n} : \mathcal{A}(Y) = \mathbf{b} \text{ and } \langle L, Y\rangle = \text{val}\}.$$

Let's denote

$$\mathcal{C}_X = \{X \in \partial\mathbb{S}^{n\times n}_+ : \mathcal{A}(X) = \mathbf{b} \text{ and } \langle L, X\rangle < \langle L, F\rangle\}.$$

Let's first show that P_F is a bijection from $\mathcal{C}_Y$ to $\mathcal{C}_X$.
Take Y_1 and Y_2 from $\mathcal{C}_Y$. Suppose $P_F(Y_1) = P_F(Y_2)$. Let $\lambda_1 = \lambda_{\min,F}(Y_1)$, $\lambda_2 = \lambda_{\min,F}(Y_2)$. Thus,

$$\frac{Y_1 - F}{1-\lambda_1} = \frac{Y_2 - F}{1-\lambda_2}. \tag{12}$$

With $\langle L, Y_1\rangle = \langle L, Y_2\rangle = \text{val}$, we can take the inner product with L then get

$$\frac{\text{val} - \langle L, F\rangle}{1-\lambda_1} = \frac{\text{val} - \langle L, F\rangle}{1-\lambda_2}$$

Since val $- \langle L, F\rangle < 0$, this shows $\lambda_1 = \lambda_2$. Then, equation (12) yields $Y_1 = Y_2$. So P_F is an injection.
Take an $X \in \mathcal{C}_X$. Let $Y = F + \frac{\langle L,F\rangle - \text{val}}{\langle L,F\rangle - \langle L,X\rangle}(X - F)$. We want to show $Y \in \mathcal{C}_Y$.

$$\langle L, Y\rangle = \langle L, F\rangle + \frac{\langle L, F\rangle - \text{val}}{\langle L, F\rangle - \langle L, X\rangle}(\langle L, X\rangle - \langle L, F\rangle) = \text{val}$$

For each $i \in [m]$, with $\text{Tr}(A_i^T F) = b_i$ and $\text{Tr}(A_i^T X) = b_i$, then

$$\text{Tr}(A_i^T Y) = \text{Tr}(A_i^T F) + \frac{\langle L, F\rangle - \text{val}}{\langle L, F\rangle - \langle L, X\rangle}(\text{Tr}(A_i^T X) - \text{Tr}(A_i^T F)) = b_i.$$

Thus, $\mathcal{A}(Y) = \mathbf{b}$. Also, since $F, X \in \mathbb{S}^{n\times n}_+$, then $Y \in \mathbb{S}^{n\times n}$. Thus, $Y \in \mathcal{C}_Y$. P_F is surjective onto set $\mathcal{C}_X$. Finally, for each $Y \in \mathcal{C}_Y$,

$$\langle L, P_F(Y)\rangle = \langle L, F\rangle + \frac{1}{1-\lambda_{\min,F}(Y)}(\text{val} - \langle L, F\rangle) \tag{13}$$

is a strictly-decreasing function with respect to $\lambda_{\min,F}(Y)$. Since $Y \to P_F(Y)$ is a bijection, solving the SDP (10) is equivalent to the eigenvalue maximization program (11). □

Now suppose Y^* solves the eigenvalue maximization program (11), by letting $Y = Y^*$ in equation 13,

$$\text{opt-val} = \langle L, P_F(Y^*)\rangle = \langle L, F\rangle + \frac{1}{1-\lambda_{\min,F}(Y^*)}(\text{val} - \langle L, F\rangle)$$

After rearranging, it turns out that

$$\lambda_{\min,F}(Y^*) = \frac{\text{val} - \text{opt-val}}{\langle L, F\rangle - \text{opt-val}}.$$

The optimization goal of solving SDP (10) is to find a feasible solution $X = P_F(Y)$ for some $Y \in \mathcal{C}_Y$ such that the objective value $\langle L, X\rangle$ is much better than $\langle L, F\rangle$, *i.e.*, for a user-chosen $\epsilon > 0$,

$$\frac{\langle L, X\rangle - \text{opt-val}}{\langle L, F\rangle - \text{opt-val}} < \epsilon. \tag{14}$$

Renegar shows that equation 14 is equivalent to

$$\lambda_{\min,F}(Y^*) - \lambda_{\min,F}(Y) \leq \frac{\epsilon}{1-\epsilon}\frac{\langle L, F\rangle - \text{val}}{\langle L, F\rangle - \text{opt-val}}. \tag{15}$$

We direct the readers to the proof in Proposition 2.4 [5]. Equation 14 and 15 shows that if $\lambda_{\min,F}(Y^*) - \lambda_{\min,F}(Y) \leq \epsilon'$ with $\epsilon' \leq \frac{\epsilon}{1-\epsilon}\frac{\langle L,F\rangle - \text{val}}{\langle L,F\rangle - \text{opt-val}}$, then

$$\frac{\langle L, X\rangle - \text{opt-val}}{\langle L, F\rangle - \text{opt-val}} < \epsilon.$$

Conversion to Standard SDP Eisenach and Liu proposes a conversion of Peng-Wei SDP into a standard form of SDP [2]. Let $I_{ab} = \frac{1}{2}(\mathbf{e}_a\mathbf{e}_b^T + \mathbf{e}_b\mathbf{e}_a^T)$ for all $a < b$, $R_a = \mathbb{1}\mathbf{e}_a^T + \mathbf{e}_a\mathbb{1}^T$. Then, we define the augmented variables

$$X' = \begin{pmatrix} X & \mathbf{0} \\ \mathbf{0} & \text{dvec}(X_{\mathcal{C}}) \end{pmatrix}, \quad I'_{ab} = \begin{pmatrix} I_{ab} & \mathbf{0} \\ \mathbf{0} & -\frac{1}{2}\text{diag}(\mathbf{e}_{ab}) \end{pmatrix},$$

where $X_{\mathcal{C}}$ is a $n \times n$ matrix of slack-variables, dvec($X_{\mathcal{C}}$) is $n^2 \times n^2$. We also define the variables R'_a, I', and L' as $(n^2+n) \times (n^2+n)$ matrices with the upper left block equal to R_a, I, and L. Then, the Peng-Wei SDP (9) can be converted into the following equivalent program:

$$\begin{aligned} \underset{X'}{\text{minimize}} \ & < L', X' > \\ \text{subject to } & \text{tr}(I'_{ab}X') = 0 \text{ for } \forall a \leq b \\ & \text{tr}(R'_aX') = 2 \text{ for } \forall a \\ & \text{tr}(I'X') = k \\ & X' \succeq 0 \end{aligned} \tag{16}$$

Now the Peng-Wei SDP is converted into the standard form introduced by Renegar in (10), by setting

$$\mathcal{A}'(Y') = (\text{Tr}(I'_{a,b}Y') \text{ for } \forall a \leq b, \text{Tr}(R'_aY') \text{ for } \forall a, \text{ and Tr}(I'Y'))$$

and

$$\mathbf{b} = [0, ..., 0, 2, ..., 2, k]^T.$$

By Theorem 1, given F strictly feasible for the Peng-Wei SDP in (9), and let val $= \langle L, F\rangle$, the SDP (16) is equivalent to

$$\begin{aligned} \underset{Y'}{\text{maximize}} \ & \lambda_{\min,F'}(Y') \\ \text{subject to } & \text{tr}(I'_{ab}Y') = 0 \text{ for } \forall a \leq b \\ & \text{tr}(R'_aY') = 2 \text{ for } \forall a \\ & \text{tr}(I'Y') = k \\ & \langle L', Y'\rangle = \text{val} \end{aligned} \tag{17}$$

Notice that we can convert the SDP (17) with $F' \neq I$ strictly feasible (I is the identity matrix in $\mathbb{S}_{(n^2+n)}$) to an equivalent optimization problem with I strictly feasible:

$$\begin{aligned} &\underset{W'}{\text{maximize}}\ \lambda_{\min}(W') \\ &\text{subject to } \text{tr}(I'_{ab}F'^{\frac{1}{2}}W'F'^{\frac{1}{2}}) = 0 \text{ for } \forall a \leq b \\ &\qquad \text{tr}(R'_aF'^{\frac{1}{2}}W'F'^{\frac{1}{2}}) = 2 \text{ for } \forall a \\ &\qquad \text{tr}(I'F'^{\frac{1}{2}}W'F'^{\frac{1}{2}}) = k \\ &\qquad \langle L', F'^{\frac{1}{2}}W'F'^{\frac{1}{2}}\rangle = \text{val} \end{aligned} \tag{18}$$

It is straightforward that W'^* is the maximizer of SDP (18) if and only if $Y' = F'^{\frac{1}{2}}W'^*F'^{\frac{1}{2}}$ is the maximizer of SDP (17). Now, let's focus on solving SDP (18) since the solution of Peng-Wei SDP is immediate once we can solve (18). By applying theorem 1 to (18), let's write down the equivalent SDP with $I \in \mathbb{S}_{(n^2+n)}$ as a strictly feasible solution

$$\begin{aligned} &\underset{Q'}{\text{minimize}}\ F'^{\frac{1}{2}}L'F'^{\frac{1}{2}}Q' \\ &\text{subject to } \text{tr}(I'_{ab}F'^{\frac{1}{2}}Q'F'^{\frac{1}{2}}) = 0 \text{ for } \forall a \leq b \\ &\qquad \text{tr}(R'_aF'^{\frac{1}{2}}Q'F'^{\frac{1}{2}}) = 2 \text{ for } \forall a \\ &\qquad \text{tr}(I'F'^{\frac{1}{2}}Q'F'^{\frac{1}{2}}) = k \\ &\qquad Q' \succeq 0 \end{aligned} \tag{19}$$

W'^* is the solution to SDP (18) if and only if $Q' = I + \frac{1}{1-\lambda_{\min}(W'^*)}(W'^* - I)$ is a solution to SDP (19), in which case

$$X'^* = F'^{\frac{1}{2}}Q'^*F'^{\frac{1}{2}} = F' + \frac{1}{1-\lambda_{\min}(W'^*)}(F'^{\frac{1}{2}}W'^*F'^{\frac{1}{2}} - F') = P_{F'}(F'^{\frac{1}{2}}W'^*F'^{\frac{1}{2}})$$

is a solution to Peng-Wei SDP.

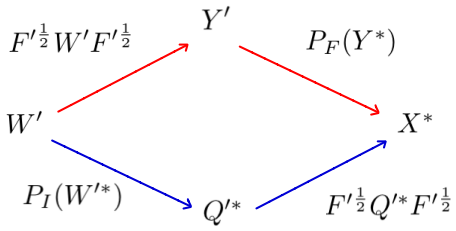

Figure 1: Schematic Diagram: two ways to solve for the maximizer X' for Peng-Wei SDP

Existence of Feasible Solutions To apply Renegar's first order method, we need a strictly feasible solution F to (16) as an input as well as a feasible solution Y_0 satisfying $\langle L, Y_0\rangle < \langle L, F\rangle$. Eisenach et al. proposes a strictly feasible solution to Peng-Wei SDP (9) and the proof of the following lemma can be found in Lemma 3 in Eisenach[2]:

Lemma 2. *Given n and k, define*

$$F_{n,k} := \frac{k-1}{n-1}I + \frac{n-k}{n^2-n}11^T. \tag{20}$$

Then, $F_{n,k}$ is strictly feasible for (9) and $||F_{n,k}^{-1}||_2 = \frac{n-1}{k-1}$.

Lemma 3. *For matrix $A \notin \mathcal{C}_\perp = \{V : \mathcal{A}(V) = 0\}$, let $\mathcal{P}_{\mathcal{C}_\perp}(A)$ denote the projection of A onto the set $\mathcal{C}_\perp$. If $\mathcal{P}_{\mathcal{C}_\perp}(A) \neq \mathbf{0}$, then $\langle \mathcal{P}_{\mathcal{C}_\perp}(A), A\rangle > 0$ and $\lambda_{max}(\mathcal{P}_{\mathcal{C}_\perp}(A)) > 0$.*

Proof. Notice that the matrix $\mathbf{0} \in \mathcal{C}_\perp$. Since $\mathcal{C}_\perp$ is a convex set,

$$\langle 0 - \mathcal{P}_{\mathcal{C}_\perp}(A), A - \mathcal{P}_{\mathcal{C}_\perp}(A)\rangle \leq 0$$

Thus,

$$0 < ||\mathcal{P}_{\mathcal{C}_\perp}(A)||^2 \leq \langle \mathcal{P}_{\mathcal{C}_\perp}(A), A\rangle$$

By definition of $\mathcal{C}_\perp$, each $V \in \mathcal{C}_\perp$ satisfying $\text{Tr}(V) = 0$. Thus, the sum of eigenvalues of V is 0. Since $\mathcal{P}_{\mathcal{C}_\perp}(A)$ is not a zero matrix, $\lambda_{\max}(\mathcal{P}_{\mathcal{C}_\perp}(A)) > 0$. □

Theorem 4. *Given a strictly feasible solution F to SDP (10), $Y_0 = F - \frac{1}{\lambda_{max,F}(\mathcal{P}_{\mathcal{C}_\perp}(L))}\mathcal{P}_{\mathcal{C}_\perp}(L)$ is a feasible solution to SDP satisfying $\langle L, Y_0\rangle < \langle L, F\rangle$.*

Proof. By Lemma 3, $\lambda_{\max}(\mathcal{P}_{\mathcal{C}_\perp}(L)) > 0$. Since $F > 0$, thus $\lambda_{\max,F}(\mathcal{P}_{\mathcal{C}_\perp}(L)) > 0$. Also, with $\langle L, \mathcal{P}_{\mathcal{C}_\perp}(L))\rangle > 0$, we have

$$\langle L, Y_0\rangle = \langle L, F\rangle - \frac{1}{\lambda_{\max,F}(\mathcal{P}_{\mathcal{C}_\perp}(L))}\langle L, \mathcal{P}_{\mathcal{C}_\perp}(L))\rangle < \langle L, F\rangle$$

Now let's show that Y_0 is feasible. Since F is feasible and $\mathcal{P}_{\mathcal{C}_\perp}(L) \in \mathcal{C}_\perp$,

$$\mathcal{A}(Y_0) = \mathcal{A}(F) + 0 = \mathbf{b}.$$

Finally we need to show $Y_0 \succeq 0$ by showing $\lambda_{\min,F}(Y_0) = 0$.

$$\begin{aligned}\lambda_{\min,F}(Y_0) &= \lambda_{\min}(F^{-\frac{1}{2}}Y_0F^{-\frac{1}{2}}) = \lambda_{\min}(I - \frac{1}{\lambda_{\max,F}(\mathcal{P}_{\mathcal{C}_\perp}(L))}F^{-\frac{1}{2}}\mathcal{P}_{\mathcal{C}_\perp}(L)F^{-\frac{1}{2}}) \\ &= 1 - \frac{\lambda_{\max}(F^{-\frac{1}{2}}\mathcal{P}_{\mathcal{C}_\perp}(L)F^{-\frac{1}{2}})}{\lambda_{\max,F}} = 0.\end{aligned}$$

Thus, $\lambda_{j,F}(Y_0) \geq 0$ for each j. Therefore, $Y_0 \succeq 0$. □

5 Algorithm

In this section, we first review Renegar's method with modification to Peng-Wei SDP, then present the necessary computational steps in applying the algorithm.

5.1 First-Order Methods with NonSmooth Scheme

Let's define an important convex set which is later used in Renegar's first order method,

$$\mathcal{C}'_\perp := \{V' \in \mathbb{S}^{(n^2+n)\times(n^2+n)} : \tilde{\mathcal{A}}'(V') = \mathbf{0}, \langle F'^{\frac{1}{2}}LF'^{\frac{1}{2}}, V'\rangle = 0\}, \tag{21}$$

where

$$\tilde{\mathcal{A}}'(V') = (\text{Tr}(I'_{a,b}F'^{\frac{1}{2}}V'F'^{\frac{1}{2}}) \text{ for } \forall a \leq b, \text{Tr}(R'_aF'^{\frac{1}{2}}V'F'^{\frac{1}{2}}) \text{ for } \forall a, \text{ and } \text{Tr}(I'F'^{\frac{1}{2}}V'F'^{\frac{1}{2}})).$$

Notice that the constraints $\text{Tr}(I'_{a,b}F'^{\frac{1}{2}}V'F'^{\frac{1}{2}}) = 0$ for $\forall a < b$ guarantees that for each V' in forms of

$$V' = \begin{pmatrix} V & \mathbf{0} \\ \mathbf{0} & \text{dvec}(V_{\mathcal{C}}) \end{pmatrix},$$

the slack variable $V_{\mathcal{C}}$ should be equal to V. Also, R'_a, I', L' are zeros everywhere except in the upper left block of size $n \times n$. Therefore, we could instead consider the following convex set

$$\mathcal{C}'_\perp := \{V' = \begin{pmatrix} V & \mathbf{0} \\ \mathbf{0} & \text{dvec}(V) \end{pmatrix}, V \in \mathbb{S}^{n\times n} : F^{\frac{1}{2}}VF^{\frac{1}{2}}\mathbb{1}_n = 0, \text{Tr}(F^{\frac{1}{2}}VF^{\frac{1}{2}}) = 0, \text{Tr}(F^{\frac{1}{2}}LF^{\frac{1}{2}}V) = 0\}. \tag{22}$$

5.1.1 Efficient calculation of the subgradient $\nabla_{W'}\lambda_{\min}(W')$

The Renegar's non-smoothed scheme internally run subgradient method, which requires the projection $G' := \nabla_{W'}\lambda_{\min}(W')$ of the gradient of the objective function in SDP (18) to the set $\mathcal{C}'_{\perp}$. Let's first find a subgradient for $\lambda_{\min}(W')$ with the following lemma:

Lemma 5. *For $W \in \mathbb{S}^+$, $\nabla_W \lambda_{min}(W) = conv\{uu^T : u^T u = 1, Wu = \lambda_{min}(W)u\}$.*

Proof. $-\lambda_{\min}$ is a convex function. The subgradient G of $-\lambda_{\min}$ at W should satisfy that for all $Z \in \mathbb{S}_n$,

$$-\lambda_{\min}(Z) \geq -\lambda_{\min}(W) + \text{Tr}(G^T(Z - W)).$$

If $G = -uu^T$, where u is the unit eigenvector corresponding to the smallest eigenvalue of Q, then

$$\text{Tr}(G^T W) = -u^T W u = -\lambda_{\min}(W), \quad -\lambda_{\min}(Z) \geq \text{Tr}(G^T Z) = -u^T Z u$$

which always hold by the definition of minimum eigenvalue. □

Exact Calculation of the subgradient Lemma 5 shows that we can get the subgradient G' by doing eigendecomposition on W' to get the eigenvector unit $u' \in \mathbb{R}^{n^2+n}$ corresponding to its smallest eigenvalue. Then, we can let $G' = u'u'^T$. Seemingly, it needs a heavy eigenvalue and eigenvector computation process since the time complexity of eigendecompostion in software such as MATLAB is $O((n^2+n)^3) = O(n^6)$. However, in fact we could utilize the special structure of W' to efficiently compute the eigenvector with respect to the smallest eigenvalue. Let's recall that if W' is feasible for SDP (18), then $W' = F^{-\frac{1}{2}} Y' F^{-\frac{1}{2}}$ where Y' is feasible for SDP(17), in particular, Y' is in form of $\begin{pmatrix} Y & \mathbf{0} \\ \mathbf{0} & \text{dvec}(Y) \end{pmatrix}$. Then

$$W' = \begin{bmatrix} F^{-\frac{1}{2}} Y F^{-\frac{1}{2}} & & \mathbf{0} & \\ & \frac{Y_{1,1}}{F_{1,1}} & & \\ \mathbf{0} & & \ddots & \\ & & & \frac{Y_{n,n}}{F_{n,n}} \end{bmatrix}$$

The eigenvalues of the diagonal matrix is each entry on the diagonal. Therefore,

$$\lambda(F'^{-\frac{1}{2}} Y' F'^{-\frac{1}{2}}) = \lambda(F^{-\frac{1}{2}} Y F^{-\frac{1}{2}}) \bigcup \{\frac{Y_{i,j}}{F_{i,j}}, i \in [n], j \in [n]\}, \tag{23}$$

Thus, the major cost of computation of $\nabla f_{\mu,F'}(Y')$ is still dominated by computing the eigenvalues of $n \times n$ matrix.

Algorithm 1: ExactMinEvec with MATLAB eig function

1. Inputs: matrix $Y \in \mathbb{S}_n$, F
2. $W \leftarrow F^{-\frac{1}{2}} Y F^{-\frac{1}{2}}$
3. (a) **If** $\lambda_{\min}(F^{-\frac{1}{2}} Y F^{-\frac{1}{2}}) \leq \min\{\frac{Y_{i,j}}{F_{i,j}}, i \in [n], j \in [n]\}$, then

$$u' = \begin{bmatrix} \mathbf{u} \\ \mathbf{0} \end{bmatrix} \text{ where } F^{-\frac{1}{2}} Y F^{-\frac{1}{2}} \mathbf{u} = \lambda_{\min}(F^{-\frac{1}{2}} Y F^{-\frac{1}{2}}) \mathbf{u}.$$

   (b) **Else** Let's denote $\tilde{i}, \tilde{j}$ as the indices when $Y_{i,j}/F_{i,j}$ achieves the minimum,

$$u' = \begin{bmatrix} 0 \cdots 1 \cdots 0 \end{bmatrix}^T \text{ where } u' \text{ is 0 everywhere except 1 at the } \tilde{i}n + \tilde{j} \text{ entry.}$$

4. Output minimum eigenvalue $\alpha = \min\{\lambda_{\min}(W), \frac{Y_{i,j}}{F_{i,j}}, i \in [n], j \in [n]\}$ and the corresponding eigenvector u'.

Approximate Calculation of the Subgradient Though we can reduce the computational complexity from $O(n^6)$ to $O(n^3)$ using the special structure of the variable in our problem, it still takes a large part in the computational complexity. Indeed, in the non-smooth scheme, we only require the eigenvector with respect to the smallest eigenvalue, rather than all the eigenvectors. Thus, an algorithm is introduced here to find out an approximation of the eigenvector efficiently.

Algorithm 2: ApproMinEvec via randomized shifted power method

1. Inputs: matrix $W \in \mathbb{S}_n$, q maximum number of iterations
   - $\sigma \leftarrow 2||W||$
   - $v \leftarrow \text{randn}(n,1)/\sqrt{n}$
2. Iteration: **for** $k = 1, 2, ..., N$
   (a) $v \leftarrow \sigma v - Wv$
   (b) $v \leftarrow \frac{v}{||v||}$
3. Output v as the approximate eigenvector, $v^T W v$ as the approximate smallest eigenvalue

The arithmetic cost of **ApproMinEvec** algorithm is $O(q)$ matrix-vector multiplication. Fact 4.1 in [6] shows that the error bounds for the algorithm, which is presented in the following Lemma:

Lemma 6. *With probability at least $1-\delta$, the **ApproMinEvec** algorithm computes a unit vector u satisfying*

$$u^T W u \leq \lambda_{min}(W) + \frac{3\epsilon}{2}||W||$$

after $q \geq \frac{1}{2} + \frac{1}{\epsilon} log(\frac{n}{\delta^2})$ iterations.

Theorem 7. *With a constant probability $1-\delta$ and a chosen $\epsilon \in (0,1)$, for W feasible for SDP (18), the **ApproMinEvec** algorithm computes a unit vector u satisfying*

$$u^T W u \leq \lambda_{min}(W) + \epsilon$$

within $O(\frac{log(n)n}{\epsilon})$ iterations.

Proof. Notice that $||W|| = ||F^{-\frac{1}{2}} Y F^{-\frac{1}{2}}|| \leq ||Y|| \cdot ||F^{-1}||$. Since Y and F are feasible for SDP (17), in particular $\text{Tr}(Y) = k$ and $Y \succeq 0$ for each iteration in the subgradient algorithm, thus $\lambda_1(Y) + ... + \lambda_n(Y) = k$, and $\lambda_1(Y) \geq ... \geq \lambda_n(Y) \geq 0$, hence

$$||Y||^2 = \text{Tr}(Y^T Y) = \sum_i \lambda_i(Y)^2 \leq (\sum_i \lambda_i(Y))^2 \leq k^2,$$

and by Lemma 2, $||F^{-1}|| = \frac{n-1}{k-1}$. Thus $||W|| \leq \frac{k(n-1)}{k-1}$. Using lemma 6, $q \geq \frac{1}{2} + \frac{3k(n-1)}{2(k-1)\epsilon} log(\frac{n}{\delta^2})$ ensures the realization of the inequality with at least $1-\delta$ probability, which completes the proof. □

5.1.2 Efficient projection onto the nullspace $\mathcal{C'}_\perp$

To ensure the efficiency of the Renegar's non-smooth algorithm, which internally runs the subgradient method with convex constraints, the projection of the gradient to the nullspace $\mathcal{C'}_\perp$ (seen in equation 22) of constraints shall be efficient. First, note that the projection of G' to $\mathcal{C'}_\perp$ is the solution to the following convex optimization problem:

$$\begin{aligned} \underset{U \in \mathbb{S}_n}{\text{minimize}} \ & \frac{1}{2}||U' - G'||_F^2 \\ \text{subject to } & F^{\frac{1}{2}} U F^{\frac{1}{2}} \mathbb{1}_n = 0 \\ & \text{Tr}(F^{\frac{1}{2}} U F^{\frac{1}{2}}) = 0 \\ & \langle F^{\frac{1}{2}} L F^{\frac{1}{2}}, U \rangle = 0 \\ & U' = \begin{pmatrix} U & \mathbf{0} \\ \mathbf{0} & \text{dvec}(U) \end{pmatrix} \end{aligned} \tag{24}$$

We solve the SDP (24) theoretically by working on its KKT condition ([1], page 267). Given a matrix G' in form of $\begin{pmatrix} G & \mathbf{0} \\ \mathbf{0} & \text{dvec}(G_{\mathcal{C}}) \end{pmatrix}$, the Lagrangian of the projection SDP (24) is

Lagrangian $\mathcal{L} : \mathbb{S}^{n\times n} \times \mathbb{R}^n \times \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ of (24) with dual variables $\lambda \in \mathbb{R}^n, v \in \mathbb{R}, w \in \mathbb{R}$ and primal variable $U \in \mathbb{S}^{n\times n}$

$$\begin{aligned}\mathcal{L}(U,\lambda,v,w) &= \frac{1}{2}||U' - G'||_F^2 + \lambda^T F^{\frac{1}{2}} U F^{\frac{1}{2}} \mathbb{1}_n + v\text{Tr}(F^{\frac{1}{2}} U F^{\frac{1}{2}}) + w\langle F^{\frac{1}{2}} L F^{\frac{1}{2}}, U\rangle \\ &= \frac{1}{2}||U - G||_F^2 + \frac{1}{2}||U - G_{\mathcal{C}}||_F^2 + \lambda^T F^{\frac{1}{2}} U F^{\frac{1}{2}} \mathbb{1}_n + v\text{Tr}(F^{\frac{1}{2}} U F^{\frac{1}{2}}) + w\langle F^{\frac{1}{2}} L F^{\frac{1}{2}}, U\rangle\end{aligned}$$

The gradient of $\mathcal{L}$ with respect to U at the solution U^* and optimal dual variables λ^*, v^*, w^* should be 0, thus

$$0 = \nabla_U \mathcal{L}(U^*, \lambda^*, v^*, w^*) = (U^* - G) + (U^* - G_{\mathcal{C}}) + F^{\frac{1}{2}} \mathbb{1}_n \lambda^{*T} F^{\frac{1}{2}} + v^* F + w^* F^{\frac{1}{2}} L F^{\frac{1}{2}},$$

$$U^* = \frac{1}{2}[G + G_{\mathcal{C}} - (F^{\frac{1}{2}} \mathbb{1}_n \lambda^{*T} F^{\frac{1}{2}} + v^* F + w^* F^{\frac{1}{2}} L F^{\frac{1}{2}})] \tag{25}$$

Moreover, U^* should satisfy $F^{\frac{1}{2}} U^* F^{\frac{1}{2}} \mathbb{1}_n = 0, \text{Tr}(FU^*) = 0, \langle F^{\frac{1}{2}} L F^{\frac{1}{2}}, U^*\rangle = 0$. Thus, by plugging equation 25 into the conditions with $F\mathbb{1}_n = \mathbb{1}_n, L\mathbb{1}_n = \mathbf{0}$, we can get

$$\begin{aligned}\mathbf{0} &= F^{\frac{1}{2}} U^* F^{\frac{1}{2}} \mathbb{1}_n = \frac{1}{2} F^{\frac{1}{2}}(G + G_{\mathcal{C}}) F^{\frac{1}{2}} \mathbb{1}_n - \frac{1}{2}(\mathbb{1}_n \lambda^{*T} \mathbb{1}_n + v^* \mathbb{1}_n + \mathbf{0}) \\ 0 &= \text{Tr}(FU^*) = \frac{1}{2}\text{Tr}(F(G + G_{\mathcal{C}})) - \frac{1}{2}[\mathbb{1}_n \lambda^{*T} + \text{Tr}(FF^T) v^* + w^* \text{Tr}(FLF)] \\ 0 &= \frac{1}{2}\langle F^{\frac{1}{2}} L F^{\frac{1}{2}}, (G + G_{\mathcal{C}})\rangle - \frac{1}{2}[0 + v^* \text{Tr}(FLF) + w^* \text{Tr}(LFLF)]\end{aligned} \tag{26}$$

We can rewrite the system of equations 26 into the matrix form:

$$\left[\begin{array}{c} F^{\frac{1}{2}}(G + G_{\mathcal{C}}) F^{\frac{1}{2}} \mathbb{1}_n \\ \hline \text{Tr}(F(G + G_{\mathcal{C}})) \\ \hline \text{Tr}(F^{\frac{1}{2}} L F^{\frac{1}{2}}(G + G_{\mathcal{C}})) \end{array}\right] = \left[\begin{array}{c|c|c} \mathbb{1}_n \mathbb{1}_n^T & \mathbb{1}_n & \mathbf{0} \\ \hline \mathbb{1}_n^T & \text{Tr}(FF^T) & \text{Tr}(FLF) \\ \hline \mathbf{0} & \text{Tr}(FLF) & \text{Tr}(LFLF) \end{array}\right] \left[\begin{array}{c} \lambda \\ \hline v \\ \hline w \end{array}\right] \tag{27}$$

The matrix in equation 27 is invariant. We only need to calculate its pseudo inverse for one time hence the projection is efficient to solve which only requires matrix multiplication. By solving the linear equation 27, we can get the dual optimal λ^*, v^* and w^*. By plugging them into equation 25, we can efficiently get U^*, and

$$\mathcal{P}_{\mathcal{C}'^{\perp}}(\nabla \lambda_{\min}(G')) = \begin{pmatrix} U^* & 0 \\ 0 & \text{dvec}(U^*) \end{pmatrix}.$$

With the methods of eigenvector computation and projection onto the nullspace, the Renegar's nonsmooth scheme in which runs the subgradient method could be implemented. I first present two subgradient methods here then show the Renegar's first order method with modification to solve Peng-Wei SDP.

The following algorithm implement **Algorithm 1** which uses MATLAB **eig** function to compute the eigenvector with respect to minimum eigenvalue accurately, but requires more computational complexity.

Algorithm 3: Subgradient Method to maximize $\lambda_{\min,F}(Y)$

1. Inputs:

   - Number of iterations: N;
   - Initial feasible solution: Y_0 satisfying $\mathcal{A}(Y_0) = \mathbf{b}$ and $\langle L, Y_0\rangle < \langle L, F\rangle$. Let val $= \langle L, Y_0\rangle$.
   - $Y = Y_0$, $\alpha = \lambda_{\min,F}(Y)$
   - $W_0 = F^{-\frac{1}{2}} Y_0 F^{-\frac{1}{2}}$
   - Distance upper bound: R, a value satisfying $||W_0 - W^*_{\text{val}}|| \leq R$

2. Iteration: **for** $k = 1, 2, ..., N$

   (a) compute a subgradient of $\nabla_W \lambda_{\min}(W_k)$ using **Algorithm 1** with Y_k, F as its input, denote G'_k as its output matrix, α_k as the minimum eigenvalue

   (b) if $\alpha_k > \alpha$: $Y \leftarrow Y_k$

   (c) compute the orthogonal projection G'_k onto the subspace $\mathcal{C}'^{\perp}$ according to equation 27 and 25

   (d) $W_{k+1} \leftarrow W_k + \frac{R}{\sqrt{N}||G_k||} G_k$, $Y_{k+1} \leftarrow F^{-\frac{1}{2}} W_{k+1} F^{-\frac{1}{2}}$

3. Output Y and terminate

The following subgradient algorithm internally runs **Algorithm 2: ApproMinEvec**, which has small computational complexity even when n is large. It also avoids the conversion between Y and W inside the algorithm, as **ApproMinEvec** takes W as its input.

Algorithm 4: Subgradient Method to maximize $\lambda_{\min,F}(Y)$

1. Inputs:

   - Number of iterations N; number of points n; number of cluster k; $\epsilon \in (0, 1)$.
   - Initial feasible solution: Y_0 satisfying $\mathcal{A}(Y_0) = \mathbf{b}$ and $\langle L, Y_0\rangle < \langle L, F\rangle$. Let val $= \langle L, Y_0\rangle$.
   - $W_0 = F^{-\frac{1}{2}} Y_0 F^{-\frac{1}{2}}$, $\alpha = \lambda_{\min}(W_0)$
   - Distance upper bound: R, a value satisfying $||W_0 - W^*_{\text{val}}|| \leq R$
   - $W \leftarrow W_0$
   - $q = \left\lceil \frac{3\log(n)k^2}{2\epsilon} \right\rceil$

2. Iteration: **for** $k = 1, 2, ..., N$

   (a) compute a subgradient of $\nabla_W \lambda_{\min}(W_k)$ using **Algorithm 2** with W_k and q as its inputs, denote G'_k as its output matrix, α_k as the minimum eigenvalue

   (b) if $\alpha_k > \alpha$: $W \leftarrow W_k$

   (c) compute the orthogonal projection G'_k onto the subspace $\mathcal{C}'^{\perp}$ according to equation 27 and 25 and denote the projection as $\tilde{G}'_k = \begin{pmatrix} \tilde{G}_k & \mathbf{0} \\ \mathbf{0} & \text{dvec}(\tilde{G}_k) \end{pmatrix}$

   (d) $W_{k+1} \leftarrow W_k + \frac{R}{\sqrt{N}||G_k||} G_k$

3. $Y \leftarrow F^{\frac{1}{2}} W F^{\frac{1}{2}}$. Output Y and terminate

Algorithm 5: Renegar's NonSmoothed Scheme for First Order Method

1. Inputs:

   - A value $0 < \epsilon < 1$, F strictly feasible for Peng-Wei SDP (9);

- Y_0' in form of $\begin{pmatrix} Y_0 & \mathbf{0} \\ \mathbf{0} & \text{dvec}(Y_0) \end{pmatrix}$ feasible for SDP (17) satisfying $\langle L', Y_0' \rangle < \langle L', F' \rangle$, $\mathcal{A}'(Y_0') = \mathbf{b}'$.
- $W_0 = F^{-\frac{1}{2}} Y_0 F^{-\frac{1}{2}}$
- Distance upper bound: diam, a value satisfying $||W_0 - W_{\text{val}}^*|| \leq \text{diam}$
- $l = -1$

2. $l + 1 \to l$, $\text{val}_l = \langle F^{\frac{1}{2}} L F^{\frac{1}{2}}, W_l \rangle$
   Apply subgradient method with inputs $W_l, R = \text{diam}, N = 9\ \text{diam}^2$, then denote V_l as its output
   if $\lambda_{\min}(V_l) \leq 1/3$: output $W = W_l$ and go to step 3
   else: Let $W_{l+1} = P_I(V_l)$ then go to step 2
3. Apply subgradient method with inputs $W, R = \text{diam}, N = \lceil 9\ \text{diam}^2/\epsilon^2 \rceil$, denote $\tilde{W}$ as its output
4. compute and output $X = F^{-\frac{1}{2}} P_I(\tilde{W}) F^{-\frac{1}{2}}$, then terminate

5.2 Renegar's Smoothed Scheme

Nesterov's Smoothing Theorem 1 shows that the SDP (10) could be transformed to an equivalent maximization problem (11) with a concave but nonsmooth objective function. To obtain faster convergence, we can apply Nesterov's smoothing technique [2] to replace the objective function in SDP (17) and SDP (18) by

$$f_{\mu,F'}(Y') := -\mu \ln \sum_j e^{-\lambda_j(F'^{-\frac{1}{2}} Y' F'^{-\frac{1}{2}})/\mu} \tag{28}$$

$$f_\mu(W') := -\mu \ln \sum_j e^{-\lambda_j(W')/\mu} \tag{29}$$

where $\mu > 0$ is a user-chosen coefficient and $\lambda_i(\cdot)$ represents the ith largest eigenvalue. The motivation to use the smoothing technique is the Lipschitz continuity of f_μ with constant $\frac{1}{\mu}$ ([5]), *i.e.*, for each W_1', W_2' in the domain of f_μ,

$$||\nabla f_\mu(W_1') - \nabla f_\mu(W_2')||_F \leq \frac{1}{\mu} |W_1' - W_2'||_F.$$

Finally, the smoothed version of (17) is

$$\begin{aligned} &\underset{Y' \in \mathbb{S}_{n^2+n}}{\text{maximize}}\ f_{\mu,F'}(Y') \\ &\text{subject to } \mathcal{A}'(Y') = \mathbf{b} \\ &\qquad\qquad \langle L', Y' \rangle = \text{val}. \end{aligned} \tag{30}$$

and the equivalent SDP with I feasible is

$$\begin{aligned} &\underset{W'}{\text{maximize}}\ f_\mu(W') \\ &\text{subject to } \text{tr}(I_{ab}' F'^{\frac{1}{2}} W' F'^{\frac{1}{2}}) = 0 \text{ for } \forall a \leq b \\ &\qquad\qquad \text{tr}(R_a' F'^{\frac{1}{2}} W' F'^{\frac{1}{2}}) = 2 \text{ for } \forall a \\ &\qquad\qquad \text{tr}(I' F'^{\frac{1}{2}} W' F'^{\frac{1}{2}}) = k \\ &\qquad\qquad \langle L', F'^{\frac{1}{2}} W' F'^{\frac{1}{2}} \rangle = \text{val} \end{aligned} \tag{31}$$

The gradient of f_μ at W' is the matrix

$$\nabla f_\mu(W') = (\sum_j e^{-\lambda_j(W')/\mu})^{-1} Q_e' \begin{pmatrix} e^{-\lambda_1(W')/\mu} & & \\ & \ddots & \\ & & e^{-\lambda_n(W')/\mu} \end{pmatrix} Q_e'^T, \tag{32}$$

where $W' = Q'_e \begin{pmatrix} \lambda_1(W') & & \\ & \ddots & \\ & & \lambda_n(W') \end{pmatrix} Q'^T_e$ is an eigendecomposition of W'. Notice that

$$\begin{aligned} \lambda(W') &= \lambda(W) \bigcup \{Y_{\mathcal{C}_{i,j}}(F_{i,j})^{-1}, i \in [n], j \in [n]\}, \\ &= \lambda(W) \bigcup \{(F^{\frac{1}{2}} W F^{\frac{1}{2}})_{i,j}(F_{i,j})^{-1}, i \in [n], j \in [n]\}, \end{aligned} \tag{33}$$

since for each Y' feasible for SDP 17, $Y = Y_{\mathcal{C}}$. The major cost of computation of $\nabla f_\mu(W')$ is still dominated by computing the eigenvalues of $n \times n$ matrix.

Renegar's Smoothed Scheme internally runs Nesterov's accelerated methods with gradient mapping. Let's first review Nesterov's accelerated method [4] and then presents Renegar's Smoothed Scheme for solving Peng-Wei SDP [5].

Algorithm 6: Nesterov's optimal method to maximize $f_\mu(W')$

1. Inputs:
   - Number of iterations: N
   - $a_0 \in (0, 1)$
   - Initial feasible variable of SDP (18): $W'_0 \in \mathbb{S}_{n^2+n}$ satisfying $\tilde{\mathcal{A}}(W'_0) = \mathbf{b}$ and $\langle F'^{\frac{1}{2}} L' F'^{\frac{1}{2}}, W'_0 \rangle < \langle F'^{\frac{1}{2}} L F'^{\frac{1}{2}}, I \rangle$. Let val $= \langle F'^{\frac{1}{2}} L' F'^{\frac{1}{2}}, W'_0 \rangle$.
   - $y_0 = W'_0$.
2. Iteration: **for** $k = 1, 2, ..., N$
   (a) compute the gradient $\nabla_{y_k} f_\mu(y_k)$ according to equation 32 and 33
   (b) compute the projection $x_{k+1} = y_k + \mu \mathcal{P}_{\mathcal{C}'^\perp}(\nabla f_\mu(y_k))$ according to equation 27
   (c) compute $a_{k+1} \in (0, 1)$ from the equation $a^2_{k+1} = (1 - a_{k+1}) a^2_k$, set $b_k = \frac{a_k(1-a_k)}{a^2_k + a_{k+1}}$, $y_{k+1} = x_{k+1} + b_k(x_{k+1} - x_k)$
3. Output x_{N+1} and terminate

Algorithm 7: Renegar's Smoothed Scheme

1. Input:
   - A value $0 < \epsilon < 1$, a feasible matrix $Y_0 \in \mathbb{S}^{n \times n}$ for Peng-Wei SDP (9) satisfying $\mathcal{A}(Y_0) = b$ and $\langle L, Y_0 \rangle < \langle L, F \rangle$
   - $W'_0 = \begin{bmatrix} F^{-\frac{1}{2}} Y_0 F^{-\frac{1}{2}} & & \mathbf{0} & \\ & \frac{Y_{1,1}}{F_{1,1}} & & \\ \mathbf{0} & & \ddots & \\ & & & \frac{Y_{n,n}}{F_{n,n}} \end{bmatrix}$
   - Let $U'_0 = I + \frac{5}{6} \frac{1}{1 - \lambda_{\min}(W'_0)} (W'_0 - I)$.
   - val$_l = \langle F'^{\frac{1}{2}} L' F'^{\frac{1}{2}}, U'_0 \rangle$
   - $\mu = \frac{1}{6 \ln n}$, $N = \lceil 12 \sqrt{\ln n} \text{diam} - 2 \rceil$, $l = -1$.
2. **Inner Iterations:**
   (a) $l = l + 1$

(b) Apply N iterations of Nesterov's first-order method to (31) with U_l' and val_l as input. Denote V_l' as the final iterate of Nesterov's first-order method

(c) **If** $\lambda_{\min}(V_l') \leq \frac{1}{3}$: output $W' = U_l'$ and go to step 3
Else: $U_{l+1}' = I + \frac{5}{6}\frac{1}{1-\lambda_{\min}(V_l')}(V_l' - I)$, $\text{val}_{l+1} = \langle F'^{\frac{1}{2}} L' F'^{\frac{1}{2}}, U_{l+1}' \rangle$ and go to step 2a.

3. Beginning at W', apply $\lceil \frac{12\sqrt{\ln n}\text{diam}}{\epsilon} - 2 \rceil$ iterations of Nesterov's first-order method to the smoothed problem (31) with $\text{val} = \langle F'^{\frac{1}{2}} L F'^{\frac{1}{2}}, W' \rangle$. Let $\tilde{W}'$ denote the output.

4. Compute and output $X' = P_F(F'^{\frac{1}{2}} \tilde{W}' F'^{\frac{1}{2}})$, then terminate.

6 Computational Complexity

6.1 Non-Smoothed Scheme

To find out the computational complexity when solving the Peng-Wei SDP, let's first review the bound of total number of iterations T of subgradient method in Theorem 4.2 of [5].

Lemma 8. *After T iterations, NonSmoothed Scheme outputs a feasible solution X' for SDP (16). Let $Q' = F'^{-\frac{1}{2}} X' F'^{-\frac{1}{2}}$, then Q' is feasible for the SDP (19) and satisfies*

$$\frac{\langle F'^{\frac{1}{2}} L' F'^{\frac{1}{2}}, Q' \rangle - \textit{opt-val}}{\langle F'^{\frac{1}{2}} L' F'^{\frac{1}{2}}, I \rangle - \textit{opt-val}} \leq \epsilon,$$

where $\textit{val}_0 := \langle F'^{\frac{1}{2}} L' F'^{\frac{1}{2}}, W_0' \rangle$ with W_0' as the input feasible matrix for SDP (18) and T is bounded by

$$(9\textit{diam}^2 + 1) \cdot [\frac{1}{\epsilon^2} + \textit{log}_{\frac{3}{2}}(\frac{\langle F'^{\frac{1}{2}} L' F'^{\frac{1}{2}}, I \rangle - \textit{opt-val}}{\langle F'^{\frac{1}{2}} L' F'^{\frac{1}{2}}, I \rangle - \textit{val}_0})].$$

Theorem 9. *The **Algorithm 5** with **ExactMinEvec** terminates after $O(\frac{n^5}{\epsilon^2})$ arithmetic operations, giving an ϵ-optimal solution; While the **Algorithm 5** with **ApproMinEvec** terminates after $O(\frac{n^3 log(n)}{\epsilon^3})$ arithmetic operations;*

Proof. Recall the definition of diam in Lemma8:

$$\text{diam} :\geq \max\{||W_x - W_y|| : W_x, W_x \in \text{Level}_{\text{val}}\} \text{ with } \text{Level}_{\text{val}} := \{W \succeq 0 : \tilde{\mathcal{A}}(W) = b, \langle F^{\frac{1}{2}} L F^{\frac{1}{2}}, W \rangle = \text{val}\}.$$

In Theorem 7, I have shown that $||W||_F \leq \frac{k(n-1)}{k-1}$ for all W feasible for SDP (18), thus,

$$\text{diam} \leq \frac{2k(n-1)}{k-1}. \tag{34}$$

In each subgradient iteration, **ExactMinEvec** require $O(n^3)$ computational complexity, and **ApproMinEvec** needs $O(\frac{\log(n)n}{\epsilon})$ computational complexity. Applying Lemma 8 and plugging in diam, it completes the proof. □

6.2 Smoothed Scheme

Renegar shows the bound of the total number of iterations of Nesterov's first order method in Theorem 7.2 in [5]. To find out the computational complexity of applying Renegar's Smoothed Scheme to solve the Peng-Wei SDP, let's first review the iteration number bound for general linear problem.

Lemma 10. *The total number of iterations of Nesterov's first order accelerated method is bounded above by*

$$12\sqrt{\textit{ln}(n)} \cdot \textit{diam} \cdot (\frac{1}{\epsilon} + \textit{log}_{\frac{5}{4}}(\frac{\langle L, F \rangle - \textit{opt-val}}{\langle L, F \rangle - \langle L, Y_0 \rangle})),$$

where Y_0 is the input matrix.

Theorem 11. *The **Algorithm 7** terminates after $O(\frac{n^4\sqrt{log(n)}}{\epsilon})$. arithmetic operations, giving an ϵ-optimal solution.*

Proof. In Theorem 9, I have shown that diam $\leq \frac{2k(n-1)}{k-1}$. For each iteration in Nesterov's accelerated method, it cost $O(n^3)$ for the eigendecomposition. Finally, with the bound of iteration numbers in Lemma 10, the total arithmetic operation for Algorithm 7 is $O(\frac{n^4\sqrt{\log(n)}}{\epsilon})$. □

7 Conclusion

In this paper, I present three efficient algorithms to solve the Peng-Wei SDP program, which is a convex relaxation of a graph cut problem and could be used to solve the partition problem of n variables. The total computational complexity of applying Renegar's Smoothed Scheme is reduced from $O(\frac{n^6}{k^2\epsilon})$ in Eisenach and Liu's **FROCE** algorithm to $O(\frac{n^4\sqrt{\log(n)}}{\epsilon})$. I also proposed two algorithms based on Renegar's Non-Smoothed Scheme, one could achieve an ϵ-optimal solution within $O(\frac{n^5}{\epsilon^2})$, the other provides a chance to solve the problem without eigen-decomposition. For the future investigation, one could try to find out an efficient rounding process to recover a feasible matrix of the Peng-Wei SDP to a matrix present graph cut. With an efficient rounding tool, it is possible to stop the algorithm even earlier.

References

[1] Stephen Boyd and Lieven Vandenberghe. Convex optimization. 2004.

[2] Carson Eisenach and Han Liu. Efficient, certifiably optimal clustering with applications to latent variable graphical models. Feb 2019.

[3] Shuyang Ling and Thomas Strohmer. Certifying global optimality of graph cuts via semidefinite relaxation: A performance guarantee for spectral clustering. *arXiv.org*, Apr 2019.

[4] Yurii Nesterov. A method for solving the convex programming problem with convergence rate O$(1/k^2)$. 1983.

[5] James Renegar. Efficient first-order methods for linear programming and semidefinite programming. *arXiv.org*, Oct 2014.

[6] Alp Yurtsever, Joel A. Tropp, Olivier Fercoq, Madeleine Udell, and Volkan Cevher. Scalable semidefinite programming. Dec 2019.