Convex and Nonsmooth Optimization
HW4: Mostly about Semidefinite Programming

Name:Zijian Liu

Spring 2020

1. (20 pts) BV Ex 5.13

   (a) Let $h(x) = (x_1(x_1-1), \cdots, x_n(x_n-1))^T$, $A = \{1, \cdots n\}$. If $S \subseteq A$, for a vector v, denote $v' = (v_{s_1}, v_{s_2}, \cdots, v_{s_{\#S}})$ as v_S, in which $s_i < s_j$ if $i < j$

$$\begin{aligned} g(\lambda,\nu) &= \inf_x L(x,\lambda,\nu) \\ &= \inf_x c^T x + \lambda^T(Ax-b) + \nu^T h(x) \\ &= \inf_x x^T diag(\nu)x + x^T(c + A^T\lambda - \nu) - \lambda^T b \\ &= \begin{cases} -\dfrac{(c+A^T\lambda-\nu)^T_{A\setminus S} diag(\nu_{A\setminus S})^{-1}(c+A^T\lambda-\nu)_{A\setminus S}}{4} - \lambda^T b \\ \qquad\qquad \exists S \subseteq A \quad s.t. \quad \nu_{A\setminus S} > 0 \quad \nu_S = (c+A^T\lambda)_S = 0 \quad (*) \\ -\infty \\ \qquad\qquad otherwise \end{cases} \end{aligned}$$

   So the Lagrange dual of this problem is:

$$\sup_{\lambda\geq 0 \quad (*)} g(\lambda,\nu) = -\frac{(c+A^T\lambda-\nu)^T_{A\setminus S} diag(\nu_{A\setminus S})^{-1}(c+A^T\lambda-\nu)_{A\setminus S}}{4} - \lambda^T b$$

   (b) Let $h(x) = (x_1(x_1-1), \cdots, x_n(x_n-1))^T$

$$\begin{aligned} g(\lambda) &= \inf_x L(x,\lambda) \\ &= \inf_x c^T x + \lambda_1^T(Ax-b) + \lambda_2^T h(x) \\ &= \inf_x x^T diag(\lambda_2)x + x^T(c + A^T\lambda_1 - \lambda_2) - \lambda_1^T b \end{aligned}$$

   If we replace λ and ν in (A) by λ_1 and λ_2, we will get the same dual problem with the same constraints. So they have the same lower bound.

2. (40 pts) Consider the primal SDP

$$\begin{gathered} \min \langle C, X\rangle \\ \text{subject to } \langle A_i, X\rangle = b_i, \quad i = 1,2 \\ X \succeq 0 \end{gathered}$$

with

$$C = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, A_1 = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}, A_2 = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}, b_1 = 1, b_2 = 0$$

(a) Does the Slater condition hold for this primal SDP, i.e., does there exist a strictly feasible $\tilde{X}$?

No. We use contradiction to proof. Suppose there exists a $\tilde{X}$ positive-definite matrix. By constraints, we know:

$$\tilde{X} = \begin{bmatrix} 1 & a \\ a & 0 \end{bmatrix}$$
$$\Rightarrow y^T \tilde{X} y = y_1^2 + 2a y_1 y_2$$

Because it is positive-definite matrix, so firstly $a = 0$. If not, $y = (b, -\frac{b}{2a})^T$ always make $y^T \tilde{X} y = 0$. But even $a = 0$, it is just semi positive-definite matrix. So this is a contradiction.

(b) What is the optimal value of the primal SDP? Is it attained, and if so, by what X?

By (a), we know there is only $X = A_1$ fitting all constraints. So the optimal of the primal SDP is $\langle C, X \rangle = 0$.

(c) Write down the dual SDP.

$$\begin{aligned} g(\Lambda, \nu) &= \inf_X \langle C, X \rangle - \langle \Lambda, X \rangle + \sum_{i=1}^{2} \nu_i (\langle A_i, X \rangle - b_i) \\ &= \inf_X \langle C - \Lambda + \sum_{i=1}^{2} \nu_i A_i, X \rangle - \nu^T b \\ &= \begin{cases} -\nu^T b & \sum_{i=1}^{2} \nu_i A_i = \Lambda - C \\ -\infty & otherwise \end{cases} \end{aligned}$$

LDP:

$$\sup_{\sum_{i=1}^{2} \nu_i A_i + C \succeq 0} g(\Lambda, \nu) = -\nu^T b$$

The dual SDP:

$$\inf_{\sum_{i=1}^{2} y_i A_i + C \succeq 0} b^T y = y_1$$

(d) Does the Slater condition hold for the dual SDP, i.e., does there exist a strictly feasible dual variable $\tilde{y}$?

Yes. For example, when $\tilde{y} = (2, 2)$, $\sum_{i=1}^{2} y_i A_i + C = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \succ 0$

(e) What is the optimal value of the dual SDP? Is it attained, and if so, by what dual variable y?

$$\sum_{i=1}^{2} y_i A_i + C \succeq 0 \Leftrightarrow \begin{bmatrix} y_1 & 1 \\ 1 & y_2 \end{bmatrix} \succeq 0$$
$$\Leftrightarrow y_1 y_2 \geq 1 \quad y_1, y_2 > 0$$

So we know:

$$\inf b^T y = 0$$

But it isn't attained.

(f) Does strong duality hold?
By (b) and (e), we know $d^* = p^*$, so strong duality holds.

(g) What can you say in general about strong duality if the Slater condition holds for at least one of a primal-dual pair of SDPs?
In this class, we have known the dual problem of a dual of SDP is itself. So if the Slater condition holds for at least one, the strong duality holds.

3. (40 pts) Exercise 5.39 on p. 285–286 in BV (see also p. 219–220). Assume that the matrix W is componentwise nonnegative, so that $W_{ij} = W_{ji}$ can be interpreted as a nonnegative weight on the edge joining vertex i to vertex j. As well as answering the questions in the exercise, also do the following. First, here is some useful information.
BV EX 5.39

(a)
- Suppose X is feasible in this problem. $X \succeq 0 \Rightarrow X = Q\Lambda Q^T$, Q is an orthogonal matrix, $\Lambda = \mathbf{diag}(\lambda_1, \cdots \lambda_n)$, λ_i is the i-th largest eigenvalue. $\mathbf{rank}\ X = 1 \Rightarrow \lambda_n > 0 = \lambda_i\ i \in \{1, \cdots, n-1\}$. Let $Q = (Q_1, \cdots, Q_n)$. So $X = Q\Lambda Q^T = \lambda_n Q_n Q_n^T = \sqrt{\lambda_n} Q_n (\sqrt{\lambda_n} Q_n)^T = xx^T$. We know $X_{ii} = 1 = x_i^2$. So $x_i = \{1, -1\}$. And know $\mathrm{tr}(WX) = \mathrm{tr}(Wxx^T) = \mathrm{tr}(x^T W x) = x^T W x$, which is same as the two-way partitioning problem.
- Suppose x is feasible in the two-way partitioning problem. Let $X = xx^T$, we know $X_{ii} = x_i^2 = 1$. $\forall y \neq \mathbf{0}$, $y^T X y = (xy)^T (xy) \geq 0 \Rightarrow X \succeq 0$. $1 \leq \mathbf{rank}\ X = xx^T \leq \min(\mathbf{rank}\ x, \mathbf{rank}\ x^T) = 1 \Rightarrow \mathbf{rank}\ X = 1$. At now $x^T W x = \mathrm{tr}(x^T W x) = \mathrm{tr}(Wxx^T) = \mathrm{tr}(WX)$, which is same as this problem.

So we know the two-way partitioning problem can be cast as the problem in this question.

(b)
- $\widehat{p}^* = \inf(\{\mathrm{tr}(WX) | X \succeq 0, X_{ii} = 1\}) = \inf(\{\mathrm{tr}(WX) | X \succeq 0, X_{ii} = 1, \mathbf{rank}\ X = 1\} \cup \{\mathrm{tr}(WX) | X \succeq 0, X_{ii} = 1, \mathbf{rank}\ X \neq 1\}) \leq \inf(\{\mathrm{tr}(WX) | X \succeq 0, X_{ii} = 1, \mathbf{rank}\ X = 1\}) = p^*$. By (a) we know $p^* = \inf(\{\mathrm{tr}(WX) | X \succeq 0, X_{ii} = 1, \mathbf{rank}\ X = 1\})$ is same as the optimal value of the two-way partitioning problem. So we know this new problem is a lower bound.
- If an optimal point X^* for this SDP has rank one, we know $\widehat{p}^* = \mathrm{tr}(WX^*) \leq p^* \leq \mathrm{tr}(WX^*) = \widehat{p}^* \Rightarrow \widehat{p}^* = p^*$, which means the lower bound is tight.

(c) Suppose $\Lambda \succeq 0$, we know the dual SDP:

$$\begin{aligned} g(\Lambda, \nu) = \inf_X L(X, \Lambda, \nu) &= \inf_X \mathrm{tr}(WX) - \langle \Lambda, X \rangle + \mathrm{tr}(\mathbf{diag}(\nu) X) - \sum_{i=1}^n \nu_i \\ &= \inf_X \mathrm{tr}((W - \Lambda + \mathbf{diag}(\nu)) X) - \mathbf{1}^T \nu \\ &= \begin{cases} -\mathbf{1}^T \nu & W - \Lambda + \mathbf{diag}(\nu) = 0 \\ -\infty & otherwise \end{cases} \end{aligned}$$

So the LDP is:

$$\sup_{W + \mathbf{diag}(\nu) \succeq 0} -\mathbf{1}^T \nu$$

We can find that the LDP of the relaxation SDP is the same one as the Lagrange dual of the two-way partitioning problem. Let d^* denote the optimal value of the Lagrange dual of the two-way partitioning problem. We know $d^* \leq \widehat{p}^* \leq p^*$($\widehat{p}^*$ and p^* have the same meaning in part (b)).

- The easiest way to set up an SDP in CVX is to use "cvx_begin sdp" instead of "cvx_begin". Then, all matrix inequalities before the next "cvx_end" will be interpreted as semidefinite inequalities. Be sure to declare any symmetric matrix variables as symmetric, like this: "variable X(n,n) symmetric". See here for more details.
- If you declare a variable as "dual variable Z" and then put ": Z" after an equality or inequality constraint, you will have access to the computed optimal dual variable for that constraint. If you want more than one dual variable, use "dual variables Z y" (no comma).
- In MATLAB, and hence also CVX, if X is a matrix, diag(X) is a vector, and if x is a vector, diag(x) is a matrix. Type "help diag" for more.
- Because the rank of a matrix is a discontinuous function, "rank(X)" is not a reliable way to find the approximate rank of a matrix, especially one that has been computed with CVX. Instead, compute the eigenvalues with "eig" (assuming X is symmetric) and estimate the rank from the eigenvalues.

(a) Using W from data set 1 and data set 2, solve the SDP given in BV (5.114) with CVX.

- For data set 1, the optimal value of (5.114) is -15.000000086649365.
- For data set 2, the optimal value of (5.114) is $-1.305511477669462e+02$.

(b) Also, solve the SDP given in BV (5.115) by CVX and compare its optimal value with the one for (5.114). For the smaller data set 1, compare the computed optimal dual variables from (5.115) with the computed optimal primal variables from (5.114) and vice versa. What are the approximate ranks of the computed optimal primal and dual matrices? Do the matrices satisfy approximate complementarity, e.g. is the matrix product $X * Z$ approximately zero?

- For data set 1, the optimal value of (5.115) is -14.999999802078772, which is larger than the optimal value of (5.114) but can be seen as the same value because of the machine precision.
- For data set 2, the optimal value of (5.115) is $-1.305511445164394e+02$, which is larger than the optimal value of (5.114) but can be seen as the same value because of the machine precision.
- For data set 1, the optimal variable ν^* of (5.114) and the optimal dual variable ν'^* and X^* of (5.115) are as following, we can see $\nu^* = \nu'^*$:

$$\nu^* = \begin{bmatrix} 3.99999869997621 \\ 2.00000353858248 \\ -0.999999978628338 \\ 2.13716619956728e-08 \\ 0.999988402743956 \\ 2.13716617736281e-08 \\ 1.00000625240751 \\ 4.00000391604524 \\ 1.99999645650641 \\ 2.00000275627257 \end{bmatrix} \qquad \nu'^* = \begin{bmatrix} 3.99999869997621 \\ 2.00000353858248 \\ -0.999999978628338 \\ 2.13716619956728e-08 \\ 0.999988402743956 \\ 2.13716617736281e-08 \\ 1.00000625240751 \\ 4.00000391604524 \\ 1.99999645650641 \\ 2.00000275627257 \end{bmatrix}$$

$$W_1 + \mathbf{diag}(\nu^*) = \Lambda^* \approx \begin{bmatrix} 5 & 0 & 1 & 1 & 2 & 1 & 1 & 2 & 1 & 0 \\ 0 & 2 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 2 & 1 & 1 & 1 & 3 & 1 & 1 & 1 & 1 & 2 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 2 & 0 & 1 & 1 \\ 2 & 1 & 1 & 1 & 1 & 1 & 0 & 4 & 2 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 2 & 3 & 0 \\ 0 & 1 & 1 & 1 & 2 & 1 & 1 & 1 & 0 & 4 \end{bmatrix}$$

More strictly, the diagonal of Λ^* is:

$$\mathbf{diag}(\Lambda^*) = \begin{bmatrix} 4.99999869997621 \\ 2.00000353858248 \\ 1.00000002137166 \\ 1.00000002137166 \\ 2.99998840274396 \\ 1.00000002137166 \\ 2.00000625240751 \\ 4.00000391604524 \\ 2.99999645650641 \\ 4.00000275627257 \end{bmatrix}$$

The eigenvalue of X^* and the dual matrices of X^* called Λ^* are:

$$eig(X^*) = \begin{bmatrix} 2.01572618194343e-09 \\ 5.59080974327843e-09 \\ 7.93943388600760e-09 \\ 1.01355090249161e-08 \\ 2.25831356163397e-08 \\ 2.80504799223204e-08 \\ 8.68842819257873e-08 \\ 1.33331213475238 \\ 1.33331213563135 \\ 7.33337556641690 \end{bmatrix} \quad eig(\Lambda^*) = \begin{bmatrix} 4.02055995180443e-09 \\ 2.13716615153410e-08 \\ 2.13716619076765e-08 \\ 0.310213037932836 \\ 1.00000100277444 \\ 1.27780965347184 \\ 2.80693892764747 \\ 3.61409306711367 \\ 5.03250470728150 \\ 11.9584396436637 \end{bmatrix}$$

So the approximate rank of rank of X^* is 3. the approximate rank of rank of Λ^* is 7. And $\langle X^*, \Lambda^* \rangle = 2.845705959209077e-07 \approx 0$. So they satisfy approximate complementarity.

(c) Here is another way to motivate the SDP relaxation (5.115). Instead of insisting that the variables x_i in (5.113) have the values ± 1, replace each scalar x_i by a vector $v_i \in \mathbb{R}^n$ with $\|v_i\|_2 = 1$, and then write $V = [v_1, \ldots, v_n]$ and $X = V^T V$. Does such an X satisfy the constraints in (5.115)? This is the motivation used in Goemans and Williamson's celebrated 1994 paper and this leads to a simple randomized procedure for assigning the vertices to the two sets: see equations (1)–(3) on p. 1120 of their paper. Solving the SDP gives you $X \succeq 0$ and then you need V such that $X = V^T V$. Is such a V unique? If not, what is a convenient choice? Show that it gives you V whose columns have norm one as required. Would this work if X is exactly low rank, or low rank to machine precision, instead of only approximately low rank? Why or why not?

Using this V, carry out the assignment algorithm on p. 1120 for the data sets 1 and 2 using r with $r_i = 1/\sqrt{n}$ (instead of a random vector) and print the resulting partitioning of the vertices and the corresponding cut value. How does it compare to the optimal value of the SDP?

- $\forall y \neq \mathbf{0}, y^T X y = y^T V^T V y = (Vy)^T (Vy) \geq 0 \Rightarrow X \succeq 0$. $X_{ii} = v_i^T v_i = \|v_i\|_2 = 1$. So X satisfy the constraints in (5.115).
- The V is not unique. If a V which satisfies $X = V^T V$. Then for any orthogonal matrices Q, QV also satisfies $(QV)^T QV = V^T Q^T QV = V^T V = X$.
- $X \succeq 0 \Rightarrow X = Q\Lambda Q^T$, Q is an orthogonal matrix, $\Lambda = \mathbf{diag}(\lambda_1, \cdots \lambda_n)$, λ_i is the i-th largest eigenvalue. So $X = Q\Lambda^{\frac{1}{2}} * (Q\Lambda^{\frac{1}{2}})^T \Rightarrow V = (Q\Lambda^{\frac{1}{2}})^T$ is a convenient choice. Suppose $V = (Q\Lambda^{\frac{1}{2}})^T = [v_1, \cdots, v_n]$. So $X = V^T V \Rightarrow X_{ii} = 1 = v_i^T v_i = \|v_i\|_2$, which means V's columns have norm one as required.
- For exactly low rank, or low rank to machine precision, it would work. Because we can always calculate the Q and Λ by function *eig*.
- – For data set 1, the cut value is -15 as same as the optimal value of SDP. The partitioning of vertices x is:

$$x = \begin{bmatrix} 1 & 1 & 1 & -1 & -1 & -1 & -1 & -1 & 1 & 1 \end{bmatrix}^T$$

  But I also try Cholesky decomposition in this problem. The answer is -11.

  – For data set 2, the cut value is -88 which is larger than the optimal value of SDP, The partitioning of vertices $x = [x_1, \cdots, x_5]^T$ is:

$$\begin{aligned} x_1 &= \begin{bmatrix} 1 & -1 & -1 & 1 & -1 & -1 & 1 & -1 & -1 & -1 \end{bmatrix} \\ x_2 &= \begin{bmatrix} -1 & -1 & -1 & -1 & 1 & -1 & 1 & 1 & 1 & -1 \end{bmatrix} \\ x_3 &= \begin{bmatrix} 1 & -1 & -1 & -1 & 1 & 1 & 1 & -1 & -1 & -1 \end{bmatrix} \\ x_4 &= \begin{bmatrix} 1 & -1 & 1 & 1 & 1 & -1 & 1 & -1 & -1 & 1 \end{bmatrix} \\ x_5 &= \begin{bmatrix} -1 & 1 & 1 & 1 & -1 & 1 & -1 & 1 & 1 & 1 \end{bmatrix} \end{aligned}$$

  But I also try Cholesky decomposition in this problem. The answer is -82.

(d) Explicitly solve (5.113) for data set 1 only. This problem is NP-hard so you will have to write a brute force method to solve it: there is no way to do it efficiently, but it should run fast enough on the smaller data set 1. According to Goemans and Williamson, the optimal value in their SDP relaxation (which CVX computes in polynomial time up to a given accuracy) should be within a factor of ≈ 0.878 of the optimal value of the max cut problem. Is it? If not, perhaps there are some issues of scaling or constants that need working through.

- The optimal value of (5.113) for data set 1 is -15, the optimal value of x is:

$$x = \begin{bmatrix} -1 & -1 & -1 & 1 & 1 & 1 & 1 & 1 & -1 & -1 \end{bmatrix}^T$$

  I think there are two points we should confirm:

  (i) The problem in the paper is not totally same as the problem in BV. We should do some calculus to find the factor.

  (ii) The factor is used for estimation of randomized SDP but not for just one times in the homework. So I try to randomize the vector r, in that case, I can also get the answer -11 which is the same answer when I used Cholesky decomposition and the fixed r.

So if we just use -11 to calculate:

$$\frac{\sum_{1\le i,j\le n} W_{ij} - (-11)}{\sum_{1\le i,j\le n} W_{ij} - (-15)} = 0.9655$$

We can find the answer satisfies the factor. If we use -15 the result is just 1, which is obviously right too. If we consider the estimation, it should be larger than 0.9655, which is also right.

An additional note of interest, not part of the homework: Håstad showed that there is no polynomial time algorithm to improve this max-cut guaranteed approximation factor from 0.878 to $16/17 \approx 0.941$ (assuming P $\neq$ NP), and Courant's Subhash Khot and his collaborators showed in this 2005 paper that if Subhash's "unique games conjecture" is true, then SDP is optimal for max-cut: one cannot get a better approximation guarantee than ≈ 0.878 in polynomial time unless P=NP. This would mean that SDP is somehow a very fundamental notion. Amazing!

Code part

main.m

```
% Read data
w1=load('hw4data1.mat').W;
w2=load('hw4data2.mat').W;
%Use cvx for (5.114)
[ds1_1,v1,~,~]=SDP(w1,1);
[ds2_1,v2,~,~]=SDP(w2,1);
%Use cvx for (5.115)
[ds1_2,X1,lambda1,nu1]=SDP(w1,2);
[ds2_2,X2,lambda2,nu2]=SDP(w2,2);
%Calculate W_1+diag(nu^*)
X3=w1+diag(v1);
%Check the result of inner product <X^*,Lambda^*>
rank_result= trace(X1'*lambda1);
%Eigendecomposition of X^* for two data sets
[Q1,L1]=eig(X1);
[Q2,L2]=eig(X2);
%Calculate V for two data sets
V1=Q1*sqrt(L1);
V2=Q2*sqrt(L2);
V1=V1';
V2=V2';
%You can choose Cholesky decomposition
% V1=chol(X1);
% V2=chol(X2);

%Calculate the vector r
[n1,~]=size(w1);
[n2,~]=size(w2);
r1=ones(n1,1)./sqrt(n1);
r2=ones(n2,1)./sqrt(n2);
%You can calculate a random vector r
% r1=2*rand(n1,1)-ones(n1,1);
% r2=2*rand(n2,1)-ones(n2,1);
```

```
%Get the resulte of inner product <r,v>
s1=r1'*V1;
s2=r2'*V2;
%Caculate how to cut
x1=ones(n1,1);
x2=ones(n2,1);
for i=1 : n1
   if s1(i)<0
       x1(i)=-1;
   end
end
for i=1 : n2
   if s2(i)<0
       x2(i)=-1;
   end
end
%Get the result of orignal problem
ds1_random=x1'*w1*x1;
ds2_random=x2'*w2*x2;
%The brute algorithm for data set 1
min=0;
xr=0;
for i=0:2^n1-1
    str = dec2bin(i);
    num=str-'0';
    num=2*num-1;
    [~,n]=size(num);
    f=ones(1,n1-n)*(-1);
    x=[f,num];
    t=x*w1*x';
    if i==0
        min=t;
        xr=x;
    elseif t<min
        min=t;
        xr=x;
    end
end
%Do some calculus for the MAX-CUT problem
sdp=(sum(sum(w1))-trace(w1))/4-(ds1_random-trace(w1))/4;
maxcut=(sum(sum(w1))-trace(w1))/4-(min-trace(w1))/4;
%Get the factor
factor=sdp/maxcut;
```

SDP.m

```
function [p,o,lambda,nu] =SDP(w,i)
[~,n] = size(w);
unit=ones(n,1);
%i=1 for (5.114)
if i==1
    cvx_begin sdp
        variable v(n)
        maximize(-unit'*v)
```

```
        w+diag(v) >= 0
    cvx_end
p = -unit'*v;
o = v;
lambda = 0;
nu = 0;
end
%i=2 for (5.115)
if i==2
    cvx_begin sdp
        variable X( n, n ) symmetric
        dual variable Y
        dual variable Z
        minimize(trace(w*X))
        unit-diag(X)== 0 : Y
        X >= 0 : Z
    cvx_end
p = sum(diag(w'*X));
o = X;
lambda = Z;
nu = Y;
end

end
```