CSCI5160 Approximation Algorithms — Spring 2022
Lecturer: Siu On Chan — *Scribe: Siu On Chan*

Notes 02: SDP vs LP; MaxCut

1. LP vs SDP

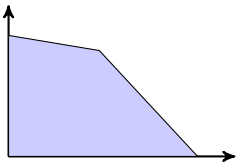
Every linear program can be represented as a semidefinite program. For example, the linear program in Eq (1) in Notes01 is equivalent to a semidefinite program with a 3-by-3 symmetric matrix X as its variables, and we may add additional constraints to force X to be diagonal (by making every off-diagonal entry to be zero). We can also make the objective matrix C diagonal, that is, the SDP objective function becomes

$$C \bullet X = \begin{pmatrix} 2 & & \\ & 3 & \\ & & -4 \end{pmatrix} \bullet \begin{pmatrix} x_1 & & \\ & x_2 & \\ & & x_3 \end{pmatrix} = 2x_1 + 3x_2 - 4x_3 \ ,$$

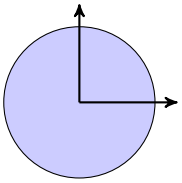
exactly the same as the linear program objective. The positive semidefinite constraint $X \succcurlyeq 0$ is equivalent to requiring all diagonal entries of X to be nonnegative, and is the same as the nonnegative constraints $x_1, x_2, x_3 \geqslant 0$ in that linear program.

In general, a semidefinite program reduces to a linear program when the objective matrix C and constraint matrices $A_1, \ldots, A_m$ are all diagonal matrices.

Recall that a linear program amounts to maximizing a linear objective function in a polyhedron (a region that is the intersection of halfspaces of the form $\{x \in \mathbb{R}^n \mid a_i^\top x \leqslant b_i\}$).



A semidefinite program amounts to maximizing a linear objective function in a feasible region whose boundary may involve curves. For example, its feasible region may be the unit circle.



This can happen with the positive semidefinite constraint

$$\begin{pmatrix} 1 & x & y \\ x & 1 & 0 \\ y & 0 & 1 \end{pmatrix} \succcurlyeq 0 \tag{1}$$

and certain matrix entries have been forced to be either 0 or 1 as indicated. Since a matrix is positive semidefinite precisely when all its principal minors are nonnegative (look up "Sylvester's criterion" on Wikipedia), (1) is equivalent to the following three constraints:

$$\begin{aligned} 1 &\geqslant 0 \\ 1 - x^2 &\geqslant 0 \\ 1 - x^2 - y^2 &\geqslant 0 \ . \end{aligned}$$

The first constraint is trivial, and the last constraint dominates the second. This shows that (1) is equivalent to (x, y) being inside the unit circle.

If a linear program involves only rational entries, then its optimal solution must also have only rational entries. The same is not necessarily true for a semidefinite program, as the previous circle example demonstrates. For instance, if we maximize along the direction $(1, 1)$ over the unit circle, the optimal solution $(1/\sqrt{2}, 1/\sqrt{2})$ has irrational entries. We cannot hope to represent a SDP optimal solution exactly, partly explaining why the algorithms for solving SDP we describe later can only look for an approximate solution.

Finally, if a linear program has finite optimum value, then the optimum value is achieved by some solution. This is again not always true for a semidefinite program. Here is a counterexample:

$$\begin{aligned} &\min \quad x \\ &\begin{pmatrix} x & 1 \\ 1 & y \end{pmatrix} \succcurlyeq 0 \end{aligned}$$

We are minimizing x, but it's equivalent to maximizing $-x$. The PSD constraint is satisfied when $x, y \geqslant 0$ and $xy \geqslant 1$. Equivalently, when $y > 0$ and $x \geqslant 1/y$. The SDP optimum value is zero, by choosing smaller and smaller $x > 0$, but $x = 0$ is not feasible. This shows that the "minimum" is in fact an infinum.

2. Max-Cut and Goemans–Williamson rounding

A classical problem is to find a subset $S \subseteq V(G)$ maximizing the number of edges going between S and $V(G) \setminus S$. This Max-Cut problem is known to be NP-hard to solve exactly, so we will settle for an approximation algorithm.

Theorem 2.1 (Goemans–Williamson [2])**.** *There is a polynomial-time algorithm to approximate Max-Cut to within* $0.878\ldots$

In other words, if the best cut in a graph with 10000 cross edges, then the algorithm will return a cut with at least 8780 cross edges. The algorithm is based on a semidefinite program first proposed by Delorme and Poljak [1]. The rounding algorithm came from a seminal work by Goemans and Williamson, where they interpret the SDP as a vector optimization program. To motivate the vector program, first formulate Max-Cut as the following quadratic integer program:

$$\begin{aligned} \max \quad & \sum_{(u,v)\in E} \frac{1 - x_u x_v}{2} \\ & x_u \in \{+1, -1\} \quad \text{for every } u \in V(G) \end{aligned}$$

Given any candidate solution S to Max-Cut, if we assign -1 to vertices in S and $+1$ to vertices outside S, then the term $(1 - x_u x_v)/2$ will be either 1 or 0, depending on whether the edge (u, v) crosses the cut. Therefore the objective function measures exactly the total number of cut edges. We are assigning ± 1 rather than $\{0, 1\}$ because it will simplify the calculations to come.

The above quadratic integer program is equivalent to Max-Cut and is NP-hard to solve exactly, so we relax the problem by instead assigning a *vector* to every vertex. We arrive at the following vector program.

$$\begin{aligned} \max \quad & \sum_{(u,v)\in E} \frac{1 - \langle y_u, y_v \rangle}{2} \\ & \|y_u\|_2^2 = 1 \quad \text{for every } u \in V(G) \\ & y_u \in \mathbb{R}^n \quad \text{for every } u \in V(G) \end{aligned} \tag{2}$$

The norm constraint $\|y_u\|_2^2 = 1$ says that we are assigning a unit vector to every vertex. Here the dimension n of the space containing the vectors equals the number of vertices of G.

Claim 2.2. *The vector program optimum value is always at least the Max-Cut optimum value.*

Proof. Pick any unit vector $y_0 \in \mathbb{R}^n$. Given a candidate solution S to Max-Cut, we assign $-y_0$ to vertices in S and y_0 to vertices outside S. Then the term $(1 - \langle y_u, y_v \rangle)/2$ will be either 1 or 0, depending on whether the edge (u, v) crosses the cut. Therefore there is always a vector assignment with value at least as large as the number of edges crossing S. □

When y_u and y_v are both unit vectors, the objective function is a sum of terms of the form

$$(1 - \langle y_u, y_v \rangle)/2 = (\|y_u\|_2^2 - 2\langle y_u, y_v \rangle + \|y_v\|_2^2)/4 = \|y_u - y_v\|_2^2/4 \,.$$

Therefore the vector program is seeking a mapping of vertices to the high dimensional unit sphere that maximizes the sum of squared distances between endpoints of an edge.

The vector program (2) is equivalent to the following semidefinite program (that can be solved efficiently):

$$\begin{aligned} \max \quad & \sum_{(u,v)\in E} \frac{1}{4} L^{uv} \bullet Y \\ & Y_{uu} = 1 \quad \text{for every } u \in V(G) \\ & Y \succcurlyeq 0 \,, \end{aligned} \tag{3}$$

where L^{uv} is the matrix that is zero everywhere, except that it is 1 at uu and vv entries, and is -1 at uv and vu entries. Recall that the PSD constraint means Y encodes the inner products between vectors $\{y_u\}_{u\in V(G)}$, so that $Y_{uv} = \langle y_u, y_v\rangle$. One can check that $L^{uv} \bullet Y/4 = \|y_u - y_v\|_2^2/4$, which equals $(1 - \langle y_u, y_v\rangle)/2$ for unit vectors y_u and y_v, so (3) and (2) have the same objective functions. The constraint $Y_{uu} = 1$ in the SDP (3) is identical to $\|y_u\|_2^2 = 1$ in the vector program (2). In general, any SDP is equivalent to a vector optimization problem with linear objective function and linear constraints on inner products.

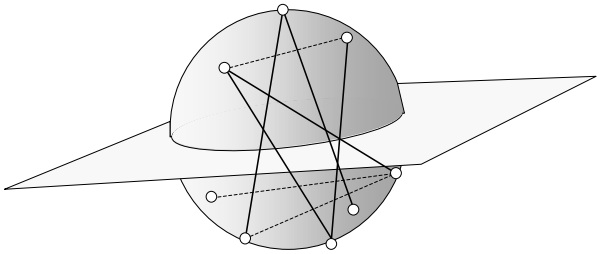


FIGURE 1. Random hyperplane rounding (picture from [3])

Given a solution to the vector program, Goemans–Williamson proposed to randomly round them into a cut, by picking a random hyperplane, and putting all vectors on one side of the hyperplane to S. In other words, one pick a random unit vector $g \in \mathbb{R}^n$ (a normal vector to the hyperplane), and set $S = \{u \in V \mid \langle y_u, g\rangle \geqslant 0\}$. We now show that, in expectation, this random subset S cuts at least $0.878\ldots$ fraction of edges in the optimal cut.

Proposition 2.3.

$$\mathbb{E}[\textit{number of edges cut}] \geqslant 0.878\cdots \times \text{SDP-OPT} \,,$$

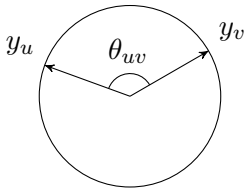
where SDP-OPT *denotes the optimum value of the SDP (or the vector program).*

Note that the proposition together with Claim 2.2 implies Theorem 2.1 (at least in expectation).

Proof of proposition. Let $\{y_u\}_{u\in V}$ be an optimal vector assignment to (2). Expand

$$\mathbb{E}[\text{number of edges cut}] = \sum_{(u,v)\in E} \mathbb{E}[\mathbb{1}((u,v) \text{ is cut by } S)]$$

by linearity of expectation. Focusing on each summand, each edge (u, v) is cut with probability θ_{uv}/π, where θ_{uv} is the angle between y_u and y_v. In other words, $\theta_{uv} = \arccos\langle y_u, y_v\rangle$.

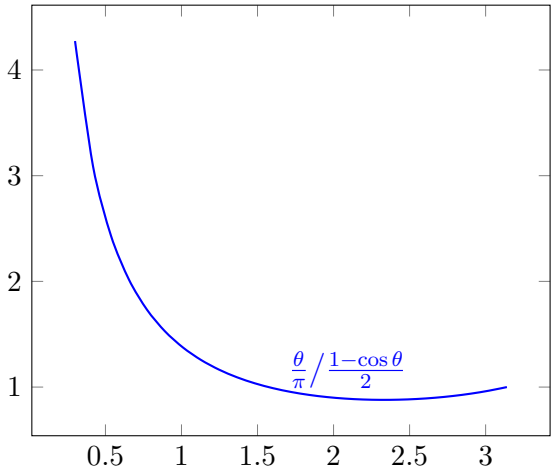


On the other hand, each edge (u, v) contributes $(1 - \langle y_u, y_v\rangle)/2 = (1 - \cos\theta_{uv})/2$ to the vector program. We now compare the contribution to the cutting probability and the contribution to the vector program.

Lemma 2.4 (Goemans–Williamson)**.** *For every* $0 \leqslant \theta \leqslant \pi$,

$$\frac{\theta}{\pi} \geqslant 0.878\cdots \times \frac{1-\cos\theta}{2}.$$

This lemma can be proved rigorously by taking derivatives. We will not do so, but instead only "prove" with a plot of $f(\theta) = \frac{\theta}{\pi}\Big/\frac{1-\cos\theta}{2}$:



Therefore

$$\begin{aligned}\mathbb{E}[\text{number of edges cut}] &= \sum_{(u,v)\in E} \mathbb{P}[(u,v) \text{ is cut by } S] = \sum_{(u,v)\in E} \frac{\theta_{uv}}{\pi} \\ &\geqslant 0.878\cdots \times \sum_{(u,v)\in E} \frac{1-\langle y_u, y_v\rangle}{2} = 0.878\cdots \times \text{SDP-OPT}. \qquad \square\end{aligned}$$

We remark that Goemans–Williamson algorithm also works with the weighted version of the problem, where each edge (u,v) has a nonnegative weight w_{uv}, and we seek to maximizing the sum of weights of edges crossing the cut.

References

[1] Charles Delorme and Svatopluk Poljak. Laplacian eigenvalues and the maximum cut problem. *Mathematical Programming*, 62(3, Ser. A):557–574, 1993.

[2] Michel Xavier Goemans and David Paul Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM*, 42(6):1115–1145, November 1995.

[3] László Lovász. Semidefinite programs and combinatorial optimization. In *Recent advances in algorithms and combinatorics*, volume 11 of *CMS Books Math./Ouvrages Math. SMC*, pages 137–194. Springer, New York, 2003.