TD9: Semidefinite Programming

Exercise 1. Positive Semidefinite Matrices
Let $X \in \mathbb{R}^{n \times n}$ be a symmetric matrix. Prove that the following statements are equivalent.

1. For all $y \in \mathbb{R}^n$, $y^\intercal X y \geq 0$.
2. All eigenvalues of X are nonnegative.
3. $\exists V \in \mathbb{R}^{m \times n}, m \leq n,$ such that $X = V^\intercal V$.

Exercise 2. Maximum Cut
In class, we presented the Goemans-Williamson approximation algorithm for the maximum cut problem on a graph $G = (V, E)$. The algorithm outputs a cut whose expected value is at least $\alpha \cdot \mathrm{O}PT$ where $\alpha \approx .87856$. Let $\varepsilon > 0$ be some small fixed positive constant and let $n = |V|$.

1. Provide a lower bound on the probability that the actual value of the cut output by the algorithm is at least $(\alpha - \varepsilon) \cdot \mathrm{O}PT$.
2. Prove that if the algorithm is run $O(\log n)$ times, then with high probability, a cut with value at least $(\alpha - \varepsilon) \cdot \mathrm{O}PT$ is found.

Exercise 3. Detecting Bipartiteness
Let $G = (V, E)$ be an unweighted, undirected graph. We wish to determine whether or not G is bipartite or not. You probably already know algorithms for this problem. Here we explore using semidefinite programming. Let $SDP(G)$ denote the value of the semidefinite programming relaxation for the maximum cut problem on graph G.

1. Suppose G is bipartite. Show that $SDP(G) = |E|$.
2. Suppose that $SDP(G) = |E|$. Prove that G is bipartite and give an algorithm to find a bipartition.

Exercise 4. Correlation Clustering with One Hyperplane
Recall the correlation clustering problem from Lecture 10. We would like to analyze the following (simpler) algorithm.

1. Choose a random vector $r = (r_1, r_2, \ldots, r_n)$ such that $r_i \sim \mathcal{N}(0, 1)$.
2. Form two clusters as follows:

$$R_1 = \{i \in V : \; r \cdot v_i \geq 0\}$$
$$R_2 = \{i \in V : \; r \cdot v_i < 0\}$$

What is the approximation guarantee of this algorithm for the correlation clustering problem?

Exercise 5. Directed Cut in Eulerian Graphs

Given a directed graph $G = (V, A)$, the *maximum directed cut* problem is to find a bipartition of the vertices $(S, V \setminus S)$ that *maximizes* the number (or weight) of edges directed from S to $V \setminus S$. Suppose that G is Eulerian. Find an β-approximation problem for the maximum directed cut problem in G for the largest value of β you can find.

Exercise 6. MAX 2-SAT

Let F be a satisfiability formula on n variables, $x_1, x_2, \ldots, x_n$, and m clauses, $C_1, \ldots, C_m$, where each C_k has one or two literals (e.g. $(x_1 \vee \neg x_2) \wedge (\neg x_1 \vee x_3) \wedge \ldots$). Each clause has a nonnegative weight w_k. The goal is to find an boolean assignment for the variables $x_1, \ldots, x_n$ so that the total weight of the satisfied clauses is maximized.

a. Show that MAX 2-SAT is NP-hard by giving a reduction from 3-SAT. Hint: Given a 3-SAT formula $F' = C'_1 \wedge C'_2 \wedge \ldots$, do the following. For each clause $C'_k = (l_1 \vee l_2 \vee l_3)$ add a variable c_k to the set of variables and add the following clauses to create a new 2-SAT formula:

$$(l_1), (l_2), (l_3), (c_k), (\neg l_1 \vee \neg l_2), (\neg l_2 \vee \neg l_3), (\neg l_1 \vee \neg l_3), (l_1 \vee \neg c_k), (l_2 \vee \neg c_k), (l_3 \vee \neg c_k)$$

b. In Lecture 8, we saw a $\frac{3}{4}$-approximation algorithm for the MAX 2-SAT problem. Now we will see how to obtain an improved approximation guarantee using SDP.

   In order to design a vector programming relaxation of this problem, we first express it as a quadratic program. Each variable x_i in F is associated with a variable $v_i \in \{1, -1\}$ (or $v_i \cdot v_i = 1$). A new variable v_0 is introduced, such that $v_0 \cdot v_0 = 1$. The variable v_0 represents the "direction of truth"; a variable x_i is interpreted as true if $v_i = v_0$. (In other words, if $v_i \cdot v_0 = 1$, then x_i is TRUE. Alternatively, if $v_i \cdot v_0 = -1$, then x_i is FALSE.) Prove that MAX 2-SAT can be expressed in this setting with an objective function Φ consisting of a sum of $a_{ij}(1 - v_i v_j)$ and $b_{ij}(1 + v_i v_j)$ for $i, j \in \{0, 1, \ldots, n\}$.

c. Relax this quadratic program into a vector program. Express the optimal solution OPT^* of your program in terms of a_{ij}, b_{ij} and $\cos \theta_{ij}$, where θ_{ij} is the angle between vectors v_i and v_j.

d. Use this vector program to give a polynomial-time algorithm to determine whether or not an instance F of 2-SAT is satisfiable.

e. Consider a random vector r and assign v_i to 1 if $r \cdot v_i > 0$ and to -1 otherwise. Express the expectation of Φ in terms of a_{ij}, b_{ij} and θ_{ij}.

f. Recall that $\alpha \sim .87856$ is the minimum of $\frac{2x}{\pi(1-\cos x)}$ when $x \in (0, \pi]$. Propose a polynomial algorithm for an α-approximation of MAX 2-SAT in expectation.