Exercises (Please Solve but Do Not Submit)

1. **Steiner Tree.** Given any Steiner tree instance (see Problem 4 below for definitions), here is a 2-approximation. Form the “metric completion” on the terminals: this is a complete graph $M = (R, (R^2))$ whose vertices are $R$, and the length of an edge $(i, j)$ is the shortest path distance between $i$ and $j$ in the original graph $G$. Compute a min-cost spanning tree (MST) $T$ on $M$. Finally, for each edge $(i, j)$ of this MST $T$, add to $T'$ all the edges on some shortest-$i$-$j$-path in $G$. Show that $T'$ is a 2-approximate Steiner tree.

2. **Multiway Cut.** The Multiway Cut takes a graph $G = (V, E)$ with a subset of terminals $S = \{s_1, s_2, \ldots, s_k\} \subseteq V$. The goal is to delete the fewest edges so that no connected component contains two different terminal nodes. Here is an algorithm: for each $i \in [1 \ldots k - 1]$, find the min-cut separating terminal $s_i$ from the set $S \setminus \{s_i\}$. (This can be solved using an $s$-$t$-min-cut algorithm.) Now take the union of these $k - 1$ cuts. Show this is a 2-approximation.

3. **Randomization and Derandomization.** Consider a 3-SAT formula $\varphi$ with $m$ clauses and $n$ variables. (We assume that each clause has exactly three literals.)

   (i) (Easy) Show that a uniformly random assignment satisfies $7/8$ of the clauses in expectation.

   (ii) How would you find such a solution deterministically? Here’s one approach, called the method of conditional expectations. A partial assignment $\pi$ is a setting of some of the $n$ variables to values in $\{T, F\}$. For a formula $\varphi$ and a partial assignment $\pi$, let $f(\varphi, \pi)$ be the expected number of clauses satisfied by setting the remaining variables (those not set by $\pi$) independently and uniformly at random. Show that we can calculate $f(\varphi, \pi)$ in linear time. (Hint: do not use sampling, even if that gives a correct but slightly slower algorithm.)

   (iii) Given any partial assignment $\pi$ and a variable $x_i$ not set by $\pi$, show that
   \[
   \max\{f(\varphi, \pi \cup \{x_i \leftarrow T\}), f(\varphi, \pi \cup \{x_i \leftarrow F\})\} \geq f(\varphi, \pi).
   \]

   (iv) Give a greedy algorithm to find a solution of value $f(\varphi, \emptyset)$. Observe this value is $\frac{7}{8}m$.

Problems

Please solve any four out of the first five problems. (The last bonus problem is optional.)
1. Local Search for Multiway Cut. The Multiway Cut takes a graph \( G = (V, E) \) with a subset of terminals \( S = \{s_1, s_2, \ldots, s_k\} \subseteq V \). The goal is to color each node in \( V \) with one of \( k \) colors such that the terminal \( s_i \) is colored with color \( i \), so as to minimize the number of bichromatic edges. (This is the same as saying: delete the fewest edges so that no connected component contains two different terminals, as in the exercise. Make sure you believe this!)

Consider a slight extension of the problem: here each vertex \( v \in V \) has an associated coloring cost function \( C_v : [k] \to \mathbb{R}_{\geq 0} \) such that the cost of coloring \( v \) with color \( i \) is \( C_v(i) \). Now we want to find a coloring \( f : V \to [k] \) so as to minimize the total cost
\[
\Phi(f) := \sum_{v \in V} C_v(f(v)) + \text{number of bichromatic edges in } f. \tag{1}
\]

Note that if we set \( C_{s_j}(i) \) to be 0 if \( i = j \) and \( \infty \) otherwise, and for each non-terminal node \( v \), we set \( C_v(i) = 0 \) for all colors \( i \), then we get back the Multiway Cut problem. In the general case, we now allow \( k \geq n \).

(i) Our local search algorithm will make moves of the following form: if we are at coloring \( f \), pick a color \( i \) and try to find the best coloring \( f' \) obtained from \( f \) by recoloring some of the vertices by the color \( i \). I.e., \( f' \) satisfies the property that either \( f'(v) = i \) or \( f'(v) = f(v) \), and it is the one with the least cost. Call a best such coloring an \( i \)-move. (In case of ties, choose one arbitrarily.) We find such an \( i \)-move later.

Show that if \( f \) is a local optimum with respect to these moves, (i.e., none of the \( k \) potential \( i \)-moves decreases the cost), then \( \Phi(f) \leq 2\Phi(OPT) \). As usual, \( OPT \) is the optimal coloring.

(ii) Since it may take a long time to reach a local minimum, change the algorithm to make a move from \( f \) to \( f' \) as long as the cost decreases by at least \( \Phi(f) \times (\varepsilon/k) \). Show that if we start from a coloring \( f_0 \), then the algorithm takes at most
\[
O\left( \frac{\log(\frac{\Phi(f_0)}{\Phi(OPT)})}{-\log(1 - \varepsilon/k)} \right) \approx O\left( \frac{k}{\varepsilon} \right) \cdot \log \left( \frac{\Phi(f_0)}{\Phi(OPT)} \right) \tag{2}
\]
local improvement steps to reach a solution of cost \( 2(1 + \varepsilon)\Phi(OPT) \).

(iii) Note that the number of steps in the above solution is not strongly polynomial: if the coloring costs \( C_v(\cdot) \) are very large, the number of rounds may be very large (albeit polynomial in the representation of the instance). One way to fix this is to choose the start state \( f_0 \) carefully. Can you show a choice of \( f_0 \) so that \( \Phi \) is at most \( \text{poly}(n, 1/\varepsilon) \)? (Showing such an answer under the assumption \( k \leq n \) will get you most of the points; for full points, your solution should work even when \( k \gg n \).)

(iv) Suppose you now wanted to make smaller local-search moves of the form: pick a vertex \( v \) and a color \( i \), and paint \( v \) with color \( i \) if the resulting \( \Phi(f) \) decreases. (These moves are called the Glauber dynamics.) True or false: all local minima of this new process are also 2-approximate. Give a proof or a counterexample.

(v) Finally, given a current coloring \( f \) and a target color \( i \), show how to find the best \( i \)-move in polynomial time? (Hint: use an \( s-t \) min-cut computation in a suitably defined graph.)
2. Amplifying Hardness by Graph Products. In class, we showed a reduction from \((1, 7/8 + \epsilon)\)-3-SAT to the Maximum Independent Set problem. In this problem, we will show how to upgrade the reduction to obtain an arbitrarily large constant factor hardness for the problem.

Let \( H \) be a graph on \([n]\). Define the \(k\)-fold product of \( H \) as the graph \( H^{\otimes k} \) whose vertices are \(k\)-tuples of vertices of \( H \) and there is an edge between \((u_1, u_2, \ldots, u_k)\) and \((v_1, v_2, \ldots, v_k)\) iff there exists an \(i\) such that \(\{u_i, v_i\}\) is an edge in \(H\).

(i) (5 points) Prove that if the largest independent set in \( H \) is of size \(\alpha n\) then the largest independent set in \( H^{\otimes k} \) is of size \(\alpha^k n^k\).

(ii) (5 points) Prove that for any constant \(C \geq 1\), there is a polynomial-time reduction from \((1, 7/8 + \epsilon)\)-3-SAT to the problem of approximating the maximum independent set within a factor \(\leq C\).

3. LP Rounding. In the \(k\)-Center problem, the instance is just like \textsc{Facility Location} and \textsc{k-Median}, but the goal is now to find \(F \subseteq V\) of size \(k\) to minimize

\[
\Phi(F) := \sum_{i \in F} f_i y_i + \max_{j \in C} d(j, F) .
\]

We will develop an LP-rounding algorithm for it. For a point \(j \in V\), define the unit ball \(B(j) := \{j' \in V \mid d(j, j') \leq 1\}\).

i To begin, suppose we know that the connection radius \(\max_j d(j, F^*)\) of the optimal solution \(F^*\) is at most exactly one. Consider the following LP:

\[
\begin{align*}
\min & \quad 1 + \sum_i f_i y_i \\
\sum_{i \in B(j)} y_i & \geq 1 \quad \forall j \in C \\
\sum_i y_i & \leq k \\
y & \geq 0.
\end{align*}
\]

Show how to find in polynomial time a solution \(F \subseteq V\) of size \(|F| \leq k\), with opening cost \(\sum_{i \in F} f_i \leq \sum_i y_i f_i\), such that \(d(j, F) \leq 3\) for each \(j \in C\). Infer that this solution \(F\) has objective function value \(\Phi(F) \leq 3\Phi(F^*)\), where \(F^*\) is the optimal solution.

ii For the previous part, we assumed the optimal connection radius equaled 1. Now give a poly-time algorithm that again has \(\Phi(F) \leq 3\Phi(F^*)\) without knowing the optimal connection radius up-front. (Hint: use the algorithm you devised above as a black box.)

4. Hardness for Steiner Tree. We now show that the Steiner tree problem is NP-hard to approximate to some constant. In this problem, we are given an graph \(G = (V, E)\) with non-negative edge-weights \(w_e\), along with a set of \textit{terminals} \(R \subseteq V\). The vertices in \(V \setminus R\) are called \textit{Steiner} nodes. The \textsc{Steiner Tree} problem asks us to find a (connected) tree \(T = (U, E')\) with \(R \subseteq U \subseteq V\) and \(E' \subseteq E\) with least weight that contains all terminals.
We use the fact that there are families of instances for Set Cover where (a) all sets have 4 elements, and (b) it is NP-hard to distinguish between instances where there exist covers with \( n/4 \) sets (YES instances), and instances where any cover uses at least \( \alpha \cdot n/4 \) sets, for some constant \( \alpha > 1 \) (NO instances).

Construct a bipartite Steiner tree instance \( G = (R \cup \{u\}, S, E) \), where nodes in \( R \) correspond to elements of the set system, nodes in \( S \) correspond to sets, and there is an edge between \( e \in R \) and \( f \in S \) if the set \( f \) contains the element \( e \). Moreover, add one more “root” node \( u \) connected to all nodes in \( S \). The terminals are \( R \cup \{u\} \), and the Steiner nodes are \( S \). All edges have length 1. Show that any Steiner tree instance arising from YES instances has a solution of cost \( n + n/4 \). Show that all solutions arising from NO instances have cost at least \( n + \alpha n/4 \). Hence, infer that it is NP-hard to approximate Steiner tree better than a factor of \((4 + \alpha)/5\).

5. SDPs for Constraint Satisfaction. In this problem, we study SDP relaxations for problems generalizing Max-Cut and Max-3-SAT, called constraint satisfaction problems. A Boolean constraint satisfaction problem consists of:

- A predicate (a.k.a. Boolean function) \( P : \{-1, 1\}^k \rightarrow \{0, 1\} \) acting on \( k \) variables.
- \( n \) Boolean variables \( x_1, x_2, \ldots, x_n \) taking value in \(-1, 1\).
- \( m \) clauses \( C_1, C_2, \ldots, C_m \) which are \( k \)-tuples of variables (these are sometimes called “constraints”, but we’ll call them “clauses” here to disambiguate from the SDP constraints).
- \( m \) negation patterns \( L_1, L_2, \ldots, L_m \in \{-1, 1\}^k \), one associated with each \( C_i \).

For any \( C_i, L_i \), we write \( L_i \cdot x_{C_i} = (L_i(1)x_{C_i(1)}, L_i(2)x_{C_i(2)}, \ldots, L_i(k)x_{C_i(k)}) \) for the \( k \)-tuple of literals associated with \( C_i, L_i \). Given such data, the goal is to find an assignment from \{-1, 1\} to \( x_i \)’s so that the fraction of literals where \( P(L_i \cdot x_{C_i}) = 1 \) is maximized. I.e., we want to maximize

\[
\max_{x \in \{-1, 1\}^n} \frac{1}{m} \sum_{i=1}^m P(L_i \cdot x_{C_i}).
\]

Here is an example: Consider the 3-SAT problem. Here, \( P : \{-1, 1\}^3 \rightarrow \{0, 1\} \) is the Boolean function that takes the value 0 if and only if all three of its inputs are 1 (and 1 otherwise). Recall: in the ±1 world, -1 is true and 1 is false. Each \( C_i \) is a tuple of the form \((u, v, w)\) and each \( L_i \) of the form \((b_u, b_v, b_w)\) and describes the clause \((b_u x_u \lor b_v x_v \lor b_w x_w)\). Notice that “negating” in the ±1-world corresponds to multiplying a variable by -1. As another example, consider the max-cut problem. Convince yourself that the associated \( P : \{-1, 1\}^2 \rightarrow \{0, 1\} \) is the “not-equal-to” predicate satisfied if and only if the two input bits are unequal. What should the clauses and negations be?

Let \( P(x) = \sum_{S \subseteq [k]} \hat{P}(S)X_S(x) \) be the Fourier polynomial representation of \( P \). Then,

\[
P(L_i \cdot x_{C_i}) = \sum_{S \subseteq [k]} \left( \hat{P}(S) \cdot \prod_{j \in S} L_i(j) \right) X_S(x) \overset{\text{def}}{=} \sum_{S \subseteq [k]} \hat{P}_{L_i}(S)X_S(x).
\]

Our SDP relaxation will have variables \( y_S \), one for every non-empty set \( S \) such that \( S \subseteq C_i \) for some \( i \leq m \). Our objective function can then be written as:

\[
\frac{1}{m} \sum_{i=1}^m \left( \hat{P}_{\emptyset} + \sum_{S \subseteq [k], S \neq \emptyset} \hat{P}_{L_i}(S) y_S \right).
\]

We will define two sets of constraints, based on the clauses and variables respectively.
1. For each $C_i$, let $M_{C_i}$ be the $2^k \times 2^k$ matrix whose rows and columns are indexed by all possible $2^k$ subsets of $C_i$. The $(S, T)$-th entry is given by $M_{C_i}(S, T) = y_S y_T$ and $M_{C_i}(S, S) = 1$.

2. Let $M_2$ be the $(n + 1) \times (n + 1)$ matrix where the first $n$ rows and columns are indexed by elements of $[n]$ and the last row and column indexed by $\emptyset$. For any $i, j$ indexing first $n$ rows and columns, $M_2(i, j) = y_{(i,j)}$ and $M_2(i, i) = 1$ for all $i$. Next, $M_2(\emptyset, \emptyset) = 1$ and finally, for any $i$, $M_2(\emptyset, i) = M_2(i, \emptyset) = y_i$.

We can now define our constraint system:

$$M_{C_i} \succeq 0 \quad \text{for all } 1 \leq i \leq m \quad \text{(Local PSDness)}$$

$$M_2 \succeq 0 \quad \text{(Global PSDness)}$$

In the following, we will see how this SDP relaxation generalizes the one we studied for Max-Cut to all constraint satisfaction problems and analyze (a special case of) Gaussian rounding.

(i) (1 point) Let $P$ be the function on 2 bits defined by $P(x_1, x_2) = 1$ if and only if $x_1 \neq x_2$. Prove that the SDP relaxation above for Max-$P$ is equivalent to the SDP relaxation for Max-Cut we studied in class.

(ii) (2 points) Write down the relaxation explicitly for $P$ defined by $P(x_1, x_2, x_3) = 1$ iff $x_1 \lor x_2 \lor x_3$ — the 3-SAT predicate — and verify that it is a valid relaxation. That is, prove that if there is an assignment $x$ that satisfies $\Delta$-fraction of the clauses of the input (exact) 3-SAT formula then the SDP optimal value is at least $\Delta$. Here, “exact” refers to all clauses being exactly on 3 literals (as opposed to $\leq 3$.)

(iii) (2 points) Let us now generalize our reasoning to all $P$. To do this, we will prove that the “local PSDness constraints” for any $C_i$ are equivalent to the existence of a probability distribution $D$ on $\{-1, 1\}^{C_i}$ (i.e. bit assignments to variables in $C_i$) such that $E_D X_S(x) = y_S$ for every $S \subseteq C_i$.

Suppose that there is a distribution $D$ on $\{-1, 1\}^{C_i}$ such that $E_D X_S(x) = y_S$. Prove that for any vector $v \in \mathbb{R}^{2^n}$, $v^T M_{C_i} v = E_D (\sum_S v_S X_S)^2$. Conclude that $M_{C_i} \succeq 0$.

(iv) Next, let’s prove the converse. If there is such a distribution for $y_S$s, then, by linearity, for every $f : \{-1, 1\}^{C_i} \rightarrow \mathbb{R}$ described by $f = \sum_{S \subseteq C_i} \hat{f}(S) X_S(x)$, $E_D f = \sum_{S \subseteq C_i} \hat{f}(S) E_D X_S(x)$. Let $p_z$ be the probability of $z \in \{-1, 1\}^{C_i}$ under $D$.

Let $f_z : \{-1, 1\}^k \rightarrow \{0, 1\}$ be the function $f_z(x)$ that takes the value 1 when $x = z$ and 0 otherwise. Set $p_z = E_D f_z$.

(v) (2 points) Prove that $\sum_{z \in \{-1, 1\}^k} p_z = 1$.

(vi) (2 points) Using that $f_z^2 = f_z$, write $f_z^2(x) = \sum_{S,T} \hat{f}_z(S) \hat{f}_z(T) X_S(x) X_T(x)$. Prove that $E_D f_z^2 = \sum_{S,T} v_S^T M_{C_i} v_T$ where $v_S$ is the vector of $2^k$ dimension indexed by subsets $S \subseteq C_i$ and $v_{S}(S) = \hat{f}_z(S)$ for every $S$. Conclude that $p_z = E_D f_z = E_D f_z^2 \geq 0$ for every $z$. Combined with the above part, conclude that $p_z$s form a probability distribution on $\{-1, 1\}^{C_i}$ as desired in part (2).

(vii) (2 points) Finally, let’s analyze the Gaussian rounding algorithm we studied in the class for the Max-$P$ problem. Suppose that for a input instance described by $(C_i, L_i)$’s, there is an SDP solution such that $M_2 = I_{n+1}$ (we call this “pairwise uniformity” of the SDP solution). Prove that Gaussian rounding (i.e. taking Cholesky factorization $M_2 = V V^T$ and setting $x_i$s to be $\text{sign}((V_i, g))$) is equivalent to outputting a random assignment.
(viii) (0 points) How good is the Gaussian rounding on pairwise uniform SDP solutions? That depends on what the SDP objective value is. For Max-Cut, if the SDP solution happens to be pairwise uniform, verify that the SDP objective value must be $\frac{1}{2}$. Thus, for Max-Cut, the approximation ratio of the algorithm in the special case when the SDP solution is pairwise uniform is 1.

6. (Bonus) SDP Integality Gaps from Pairwise Uniformity. Unlike Max-Cut (and more generally, 2-bit predicates $P$), for an appropriate class of 3-bit predicates $P$, one can construct SDP solutions that are 1) pairwise uniform and 2) the SDP objective value is 1. In this problem, we will prove something even stronger – that there are instances of 3-SAT that admit 1) a pairwise uniform SDP solution such that 2) the SDP objective value is 1 but 3) no assignment satisfies more than a $7/8$-fraction of the constraints. This immediately shows that Gaussian rounding studied in previous problem for 3-SAT can not have a better approximation ratio than returning a random assignment and that moreover, the SDP above has an integrality gap of $7/8$ for Max-3-SAT! One can use this gap instance to construct a dictator test and obtain a UGC based (and with more work, remove the dependence on the truth of the UGC) hardness of approximation for 3-SAT.

We use the setup from the previous problem here. Moreover, the clauses $C_1, C_2, \ldots, C_m$ of an instance of Max-P are called essentially disjoint if for every $i \neq j$, $|C_i \cap C_j| \leq 1$. That is, the “clauses” intersect in at most 1 variable. A predicate $P$ is called pairwise uniform if there is a probability distribution $D_P$ on $\{-1,1\}^k$ such that for every $x$ in the support of $D_P$, $P(x) = 1$ while $E_{x \sim D_P}x_i = 0$ and $E_{x \sim D_P}x_ix_j = 0$ for every $i,j \leq k$.

Consider any instance of the Max-P problem where 1) the clauses are essentially disjoint and 2) $P$ is pairwise uniform. We show how to construct a pairwise uniform SDP solution for it with SDP objective value 1 (regardless of the negation patterns $L_i$s).

(i) Consider the following SDP solution: Set $y_i = 0$, $y_{i,j} = 0$ for every $1 \leq i,j \leq n$ and for every $S \subseteq C_i$, set $y_S = E_{x \sim D \circ L_i} \prod_{x \in S} x_i$: here $x \sim D \circ L_i$ means draw $x \sim D$ and output $(x_{C_i(1)}L_i(1), x_{C_i(2)}L_i(2), \ldots, x_{C_i(k)}L_i(k))$. Prove that the $y_S$ above form 1) a feasible solution to the SDP and 2) the objective value of the solution is 1. (Hint: Prove that for any $C_i$, $\sum_{S \subseteq C_i} P_i(S)y_S = E_{D \circ L_i} P(x_{C_i(1)}L_1, \ldots, x_{C_i(k)}L_i(k))$ and observe that the RHS is 1).

(ii) Show that for any fixed positive integer $B > 0$, for $n$ large enough, there is a collection of $m = Bn$ triples $C_1, C_2, \ldots, C_m$ on $n$ variables such that $|C_i \cap C_j| \leq 1$ for every $i \neq j$. (Hint: Choose $C_i$s uniformly at random iteratively and throw away any $C_i$ that intersects any previous constraint in $\geq 2$ positions. Argue by union bound that the chance that you throw away a pair is small if the number of steps in the iteration is $\leq m = Bn$.)

(iii) For any $\delta > 0$, let $C_1, C_2, \ldots, C_m$ for $m = B_\delta n$ where $B_\delta$ is a constant depending only on $\delta$ be a collection of triples of $n$ variables. For each $C_i$, choose a uniformly random negation pattern. Prove that no assignment satisfies more than $7/8 + \delta$ fraction of constraints the resulting 3-SAT instance (it is okay to prove this for a larger constant than 1000). (Hint: fix an assignment $x$. Over the randomness of the negation patterns, what is the probability that $x$ satisfies $7/8 + \eta$ fraction of the clauses? Now do a union bound. If you haven’t encountered it yet, you may want to look up Chernoff bounds.)

(iv) Prove that there is a probability distribution $D$ on $\{-1,1\}^3$ such that 1) for every point $x$ in the support of $D$, at least one of the $x_i$s is 1 and 2) $E_D x_i = 0$ for every $i \leq 3$ and $E_D x_i x_j = 0$ for every $i,j \leq 3$. 

6
(v) Infer from the parts above that if we take an essentially disjoint collection of $C_i$s with \( \geq B_\delta n \) clauses, randomly choose the negation patterns independently for each $C_i$ then for the resulting (random) 3-SAT formula 1) no assignment satisfies more than \( > \frac{7}{8} + \delta \) fraction of constraints with probability at least 0.99 while 2) regardless of the negation patterns, there is an SDP solution (to the SDP above) with objective value 1.

(vi) Let $P$ be the “$\neq$” predicate. Is $P$ pairwise uniform?