
Problem 1. Let H be a set of halfplanes in $\mathbb{R}^2$. Prove: H is infeasible for LP if and only if H has 3 halfplanes (not necessarily distinct) with an empty intersection.

Proof. The *if* direction is easy to prove by noting that $\emptyset \cap H = \emptyset$. Therefore, the intersection of all halfplanes must be empty and thus H is infeasible.

The *only-if* direction: Define $\cap H_i = \bigcap_{k=1}^{i} H_k$. Since H is infeasible for LP, there must exist $k \geq 2$ where $\cap H_k$ is infeasible while $\cap H_{k-1}$ is feasible. Let $\hat{n}$ be the normal vector defining H_k, and points toward the interior of H_k. Let p be the farthest point in $\cap H_{k-1}$ along the direction of $\hat{n}$. If p does not exist (implying that $\cap H_{k-1}$ is unbounded) or $p \in H_k$, $\cap H_k$ must be feasible, a contradiction. Otherwise, if $p \notin H_k$, since p must be the intersection of two boundaries of halfplanes, it must hold that the two halfplanes with H_k altogether constitute an empty intersection. □

Problem 2. Extend our 2D LP algorithm to handle also the case where the input set H of halfplanes is infeasible.

Solution. **Lemma:** Let $1 \leq i \leq n$, and let $\cap H_i = \{M_1, M_2, H_1, H_2, \ldots, H_i\}$ and v_i be the optimal vertex computed by the algorithm after processing $\cap H_i$.

1. If $v_{i-1} \in H_i$, then $v_i = v_{i-1}$;
2. If $v_{i-1} \notin H_i$, then either $\cap H_i = \emptyset$ or $v_i \in \ell_i$, where ℓ_i is the line bounding H_i.

Therefore, we just need to consider the case where $v_{i-1} \notin H_i$. If C_i is infeasible, it must follow that $v_i \notin \ell_i$, and therefore we can also project the n halfplanes on h_i and cast it into a 1D problem solvable in $O(n)$ time, determining if such v_i exists and report infeasible if no v_i exists. The program terminates with any infeasibility detected.

Problem 3. In the lecture, we proved that our 2D LP algorithm runs in $O(n)$ expected time when the input set H of halfplanes is feasible. Extend the proof to the case where H is infeasible. You can still make the general position assumption that no three planes' boundary lines pass the same point.

Proof. Suppose the algorithm has come to a point where $\cap H_i = \emptyset$. Note that $\cap H_{i-1} \neq \emptyset$ since the algorithm would have terminated if this is the case. To bound $\mathbb{E}[\mathcal{X}_i]$, fix the subset of the first i halfplanes. We prove that $\mathbb{E}[\mathcal{X}_i] = O(1/i)$ as follows.

(i) Suppose there is only one set of halfplanes $\{H_i, H_j, H_k\}$ with size of 3, forming an empty intersection, then all 3 of these halfplanes are choices for forming an infeasible region at this moment, $\mathbb{P}(\mathcal{X}_i = 1) \leq 3/i$.

(ii) If there exists any two set of halfplanes $\{H_i, H_j, H_k\}, \{H_i, H'_j, H'_k\}$ with $H_j \neq H'_j, H_k \neq H'_k$, we are certain that H_k is the only choice, $\mathbb{P}(\mathcal{X}_i = 1) = 1/i$; Otherwise, suppose WLOG that it is $H_j = H'_j$, then H_i, H_j are the possible choices, $\mathbb{P}(\mathcal{X}_i = 1) = 2/i$.

□

Problem 4. Prove: our 2D LP algorithm runs in $O(n)$ expected time even if the boundary lines of three (or more) halfplanes pass the same point. You can focus on the case where the problem is feasible.
Hint: Let us consider the processing of the last plane in H. We proved that we had to spend $O(n)$ time on this plane with a probability at most $2/n$. Show that this is the case no matter how many planes' boundary planes pass the same point.

Proof. If v_i is passed by multiple lines, then it has to be that $\mathcal{X}_i = 0$, since there are at least 2 lines left defining v_i. Therefore, it returns to the general case that $\mathbb{E}[\mathcal{X}_i] = 2/i$ for which v_i is defined by two halfplanes. □

Problem 5. Give an algorithm to solve LP in $O(n)$ expected time in 3D space.

Proof. For simplicity, we are only required to solve the problem with objective function be $z = -y$ (denote this by a vector $\hat{\mathbf{v}}$ pointing to the direction maximizing the function).

procedure 3DLinearProgramming($H_1, H_2, \ldots, H_n$)
  Randomly permutate the n halfplanes by RandomPermutation($H_1, H_2, \ldots, H_n$)
  First bound the LP by adding $M_1 : x \le m_1, M_2 : y \ge m_2, v_0 \leftarrow (m_1, m_2)$
  for $i \leftarrow 1$ to n **do**
    if $v_{i-1} \in H_i$ **then**
      $v_i \leftarrow v_{i-1}$
    else
      Solve the 2D instance formed by projecting $M_1, M_2, H_1, H_2, \ldots, H_n$ and $\hat{v}$ onto ℓ_i
    end if
  end for
  return (p', p'')
end procedure

The proof for Problem 3 applies to any dimension d. Problem 4 also implies that v_i under the constraints of $\cap H_i$ cannot have $\mathcal{X}_i = 1$ and passed by multiple lines at the same time. Therefore, we are able to apply the general assumption in the analysis. For the case when $v_i \neq v_{i-1}, i \ge d$, apply backward analysis and observe that v_i changes when we remove a halfplane that defines v_i. This can only happen for d hyperplanes under the general assumption, and thus $\mathbb{E}[\mathcal{X}_i] = \mathbb{P}(\mathcal{X}_i = 1) \le d/i$.

The time complexity of the algorithm is given by

$$\begin{aligned} T_d(n) &= O(dn) + O(dn) + O(dT_{d-1}(n)) + \sum_{i=d+1}^{n} (O(di) + T_{d-1}(i-1)) \cdot \mathbb{E}[\mathcal{X}_i] \\ &= O(dn) + O(dC_{d-1}n) + O(dn) + O(C_{d-1}n \sum_{i=d+1}^{n} O(1/i)) \\ &= O(dC_{d-1}n) \end{aligned}$$

where the two $O(dn)$ comes from randomization and testing whether $v_{i-1} \in H_i$, we can also start by solving the $d-1$ LPs of dimension $d-1$. This takes at most $dT_{d-1}(n)$ time. The last term corresponds to the cost due to $v_{i-1} \notin H_i$.

This forms a inductive proof and gives $T_d(n) = O(d!n)$ in expectation. For this problem where $d = 3$, this is expected $O(n)$ as required. □

Problem 6. A polygon is *star-shaped* if there is a point p inside the polygon that is visible to all the vertices of the polygon (two points in a polygon are visible to each other if the segment connecting the two points is completely inside the polygon). Given a polygon of n vertices, determine in $O(n)$ expected time whether it is star-shaped.
Hint: Cast this into an LP problem.

Solution. Each edge of a polygon defines an interior half-plane, the half-plane whose boundary lies on the line containing the edge and that contains the points of the polygon in a neighborhood of any interior point of the edge. Thus, the intersection of all halfplanes formed by n vertices is a set of possible vertices visible to all vertices of the polygon.

The algorithm starts with recognizing all n halfplanes and checks the feasibility of the linear program afterwards.

Start at any vertex, the interior of the polygon always lies on the right hand side of the observer when he traverse in clockwise order along the boundary of the polygon. For every adjacent pair of vertex p_i, p_j (where p_i is the vertex visited first by the observer), it defines a halfplane with the form of $\overrightarrow{p_ip_j} \times \overrightarrow{p_ip} \leq 0$ (turning clockwise into the interior). This process constructs n halfplanes in $O(n)$ time.

The LP generated can be solved in $O(n)$ expected time as required.