Chapter 54

Low Dimensional Linear Programming

By Sariel Har-Peled, April 26, 2022[①]

> "Napoleon has not been conquered by man. He was greater than all of us. But god punished him because he relied on his own intelligence alone, until that prodigious instrument was strained to breaking point. Everything breaks in the end."
> – Carl XIV Johan, King of Sweden.

54.1. Linear programming in constant dimension ($d > 2$)

Let assume that we have a set H of n linear inequalities defined over d (d is a small constant) variables. Every inequality in H defines a closed half space in $\mathbb{R}^d$. Given a vector $\overrightarrow{c} = (c_1, \ldots, c_d)$ we want to find $p = (p_1, \ldots, p_d) \in \mathbb{R}^d$ which is in all the half spaces $h \in H$ and $f(p) = \sum_i c_i p_i$ is maximized. Formally:

LP in d dimensions:$(H, \overrightarrow{c})$
H - set of n closed half spaces in $\mathbb{R}^d$
$\overrightarrow{c}$ - vector in d dimensions
Find $p \in \mathbb{R}^d$ s.t. $\forall h \in H$ we have $p \in h$ and $f(p)$ is maximized.
Where $f(p) = \langle p, \overrightarrow{c} \rangle$.

A closed half space in d dimensions is defined by an inequality of the form

$$a_1x_1 + a_2x_2 + \cdots + a_nx_n \leq b_n.$$

One difficulty that we ignored earlier, is that the optimal solution for the LP might be unbounded, see Figure 54.1.

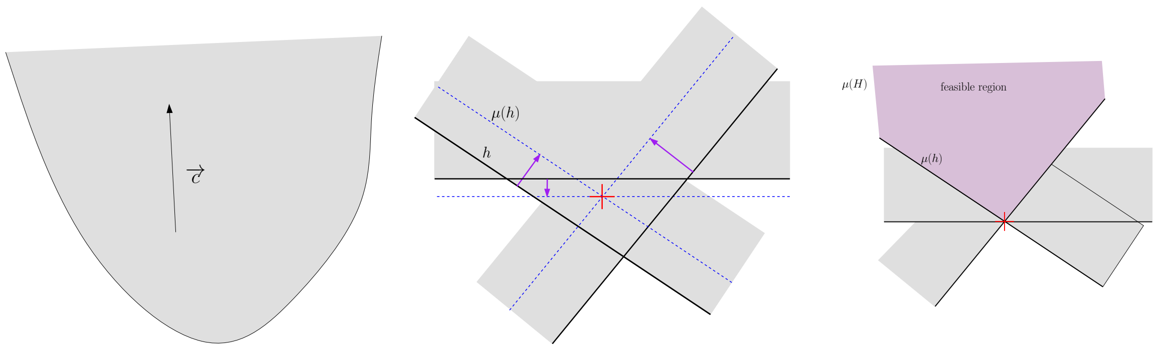
Namely, we can find a solution with value ∞ to the target function.

For a half space h let $\eta(h)$ denote the normal of h directed into the feasible region. Let $\mu(h)$ denote the closed half space, resulting from h by translating it so that it passes through the origin. Let $\mu(H)$ be the resulting set of half spaces from H. See Figure 54.1 (b).

The new set of constraints $\mu(H)$ is depicted in Figure 54.1 (c).

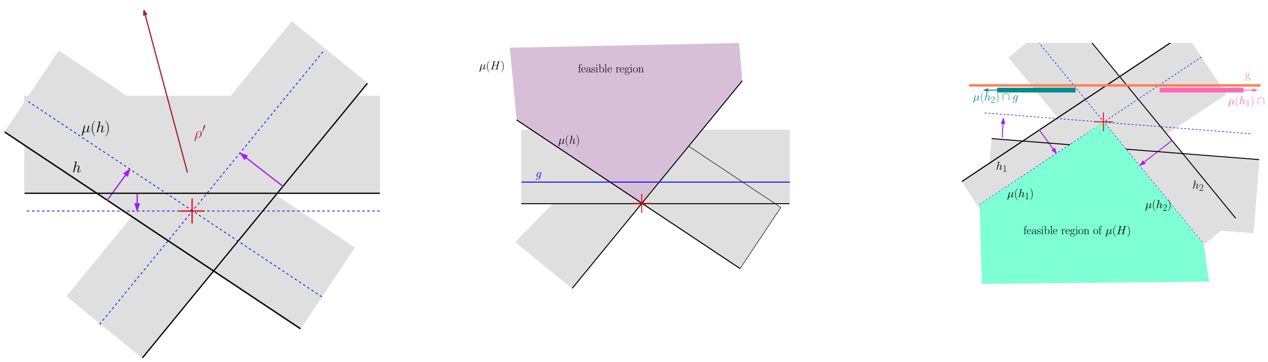
Lemma 54.1.1. $(H, \overrightarrow{c})$ *is unbounded if and only if* $(\mu(H), \overrightarrow{c})$ *is unbounded.*

Proof: Consider the ρ' the unbounded ray in the feasible region of $(H, \overrightarrow{c})$ such that the line that contain it passes through the origin. Clearly, ρ' is unbounded also in $(H, \overrightarrow{c})$, and this is if and only if. See Figure 54.2 (a). ■



(a) (b) (c)

Figure 54.1: (a) Unbounded LP. (b). (c).



(a) (b) (c)

Figure 54.2: (a). (b). (c).

Lemma 54.1.2. *Deciding if $(\mu(H), \overrightarrow{c})$ is bounded can be done by solving a $d - 1$ dimensional LP. Furthermore, if it is bounded, then we have a set of d constraints, such that their intersection prove this.*

Furthermore, the corresponding set of d constraints in H testify that $(H, \overrightarrow{c})$ is bounded.

Proof: Rotate space, such that $\overrightarrow{c}$ is the vector $(0, 0, \ldots, 0, 1)$. And consider the hyperplane $g \equiv x_d = 1$. Clearly, $(\mu(H), \overrightarrow{c})$ is unbounded if and only if the region $g \cap \bigcap_{h \in \mu(H)} h$ is non-empty. By deciding if this region is unbounded, is equivalent to solving the following LP: $L' = (H', (1, 0, \ldots, 0))$ where

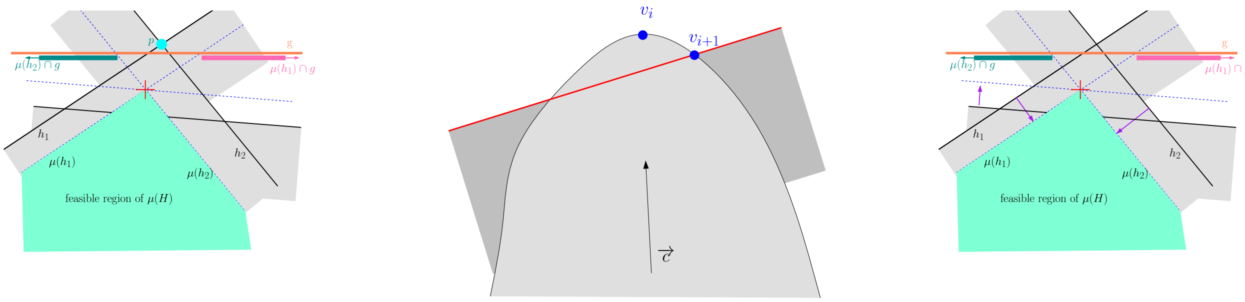
$$H' = \left\{ g \cap h \;\middle|\; h \in \mu(H) \right\}.$$

Let $h \equiv a_1 x_1 + \ldots + a_d x_d \leq 0$, the region corresponding to $g \cap h$ is $a_1 x_1 + \cdots + a_{d-1} x_{d-1} \leq -a_d$ which is a $d - 1$ dimensional hyperplane. See Figure 54.2 (b).

But this is a $d - 1$ dimensional LP, because everything happens on the hyperplane $x_d = 1$.

Notice that if $(\mu(H), \overrightarrow{c})$ is bounded (which happens if and only if $(H, \overrightarrow{c})$ is bounded), then L' is infeasible, and the LP L' would return us a set d constraints that their intersection is empty. Interpreting those constraints in the original LP, results in a set of constraints that their intersection is bounded in the direction of $\overrightarrow{c}$. See Figure 54.2 (c).

(In the above example, $\mu(H) \cap g$ is infeasible because the intersection of $\mu(h_2) \cap g$ and $\mu(h_1) \cap g$ is empty, which implies that $h_1 \cap h_2$ is bounded in the direction $\overrightarrow{c}$ which we care about. The positive y direction in this figure.) ■



(a) (b) (c)

Figure 54.3: (a). (b). (c).

We are now ready to show the algorithm for the LP for $L = (H, \overrightarrow{c})$. By solving a $d-1$ dimensional LP we decide whether L is unbounded. If it is unbounded, we are done (we also found the unbounded solution, if you go carefully through the details).

See Figure 54.3 (a).

(in the above figure, we computed p.)

In fact, we just computed a set $h_1, \ldots, h_d$ s.t. their intersection is bounded in the direction of $\overrightarrow{c}$ (thats what the boundness check returned).

Let us randomly permute the remaining half spaces of H, and let $h_1, h_2, \ldots, h_d, h_{d+1}, \ldots, h_n$ be the resulting permutation.

Let v_i be the vertex realizing the optimal solution for the LP:

$$L_i = \left(\{h_1, \ldots, h_i\}, \overrightarrow{c}\right)$$

There are two possibilities:

1. $v_i = v_{i+1}$. This means that $v_i \in h_{i+1}$ and it can be checked in constant time.

2. $v_i \neq v_{i+1}$. It must be that $v_i \notin h_{i+1}$ but then, we must have... What is depicted in Figure 54.3 (b).

B - the set of d constraints that define v_{i+1}. If $h_{i+1} \notin B$ then $v_i = v_{i+1}$. As such, the probability of $v_i \neq v_{i+1}$ is roughly d/i because this is the probability that one of the elements of B is h_{i+1}. Indeed, fix the first $i+1$ elements, and observe that there are d elements that are marked (those are the elements of B). Thus, we are asking what is the probability of one of d marked elements to be the last one in a random permutation of $h_{d+1}, \ldots, h_{i+1}$, which is exactly $d/(i+1-d)$.

Note that if some of the elements of B is $h_1, \ldots, h_d$ than the above expression just decreases (as there are less marked elements).

Well, let us restrict our attention to ∂h_{i+1}. Clearly, the optimal solution to L_{i+1} on h_{i+1} is the required v_{i+1}. Namely, we solve the LP $L_{i+1} \cap h_{i+1}$ using recursion.

This takes $T(i+1, d-1)$ time. What is the probability that $v_{i+1} \neq v_i$?

Well, one of the d constraints defining v_{i+1} has to be h_{i+1}.The probability for that is ≤ 1 for $i \leq 2d-1$, and it is

$$\leq \frac{d}{i+1-d},$$

otherwise.

Summarizing everything, we have:

$$\begin{aligned} T(n,d) &= O(n) + T(n,d-1) + \sum_{i=d+1}^{2d} T(i,d-1) \\ &+ \sum_{i=2d+1}^{n} \frac{d}{i+1-d} T(i,d-1) \end{aligned}$$

What is the solution of this monster? Well, one essentially to guess the solution and verify it. To guess solution, let us "simplify" (incorrectly) the recursion to :

$$T(n,d) = O(n) + T(n,d-1) + d \sum_{i=2d+1}^{n} \frac{T(i,d-1)}{i+1-d}$$

So think about the recursion tree. Now, every element in the sum is going to contribute a near constant factor, because we divide it by (roughly) $i+1-d$ and also, we are guessing the the optimal solution is linear/near linear.

In every level of the recursion we are going to penalized by a multiplicative factor of d. Thus, it is natural, to conjecture that $T(n,d) \leq (3d)^{3d} n$.

Which can be verified by tedious substitution into the recurrence, and is left as exercise.

Theorem 54.1.3. *Given an d dimensional LP $(H, \vec{c})$,it can be solved in expected $O\left((3d)^{3d} n\right)$ time (the constant in the O is dim independent).*

BTW, we are being a bit conservative about the constant. In fact, one can prove that the running time is $d!n$. Which is still exponential in d.

```
SolveLP((H, c⃗))
    /* initialization */
    Rotate (H, c⃗) s.t. c⃗ = (0, ..., 1)
    Solve recursively the d − 1 dim LP:
            L′ ≡ μ(H) ∩ (x_d = 1)
    if L′ has a solution then
        return "Unbounded"

    Let g_1, ..., g_d be the set of constraints of L′ that testifies that L′ is infeasible
    Let h_1, ..., h_d be the hyperplanes of H corresponding to g_1, ..., g_d
    Permute H s.t. h_1, ..., h_d are first.
    v_d = ∂h_1 ∩ ∂h_2 ∩ ··· ∩ ∂h_d
    /*v_d is a vertex that testifies that (H, c⃗) is bounded */

    /* the algorithm itself */
    for i ← d + 1 to n do
      if v_{i−1} ∈ h_i then
          v_i ← v_{i−1}
      else
          v_i ← SolveLP((H_{i−1} ∩ ∂h_i, c⃗))        (*)
              where H_{i−1} = {h_1, ..., h_{i−1}}

     return v_n
```

54.2. Handling Infeasible Linear Programs

In the above discussion, we glossed over the question of how to handle LPs which are infeasible. This requires slightly modifying our algorithm to handle this case, and I am only describing the required modifications.

First, the simplest case, where we are given an LP L which is one dimensional (i.e., defined over one variable). Clearly, we can solve this LP in linear time (verify!), and furthermore, if there is no solution, we can return two input inequality $ax \leq b$ and $cx \geq d$ for which there is no solution together (i.e., those two inequalities [i.e., constraints] testifies that the LP is not satisfiable).

Next, assume that the algorithm `SolveLP` when called on a $d-1$ dimensional LP L', if L' is not feasible it return the d constraints of L' that together have non-empty intersection. Namely, those constraints are the witnesses that L' is infeasible.

So the only place, where we can get such answer, is when computing v_i (in the (*) line in the algorithm). Let $h'_1, \ldots, h'_d$ be the corresponding set of d constraints of H_{i-1} that testifies that $(H_{i-1} \cap \partial h_i, \overrightarrow{c})$ is an infeasible LP. Clearly, $h'_1, \ldots, h'_d, h_i$ must be a set of $d+1$ constraints that are together are infeasible, and that is what `SolveLP` returns.

54.3. References

The description in this class notes is loosely based on the description of low dimensional LP in the book of de Berg *et al.* [BCKO08].

References

[BCKO08] M. de Berg, O. Cheong, M. J. van Kreveld, and M. H. Overmars. *Computational geometry: algorithms and applications*. 3rd. Santa Clara, CA, USA: Springer, 2008.