2
Linear Programming

2.1 Introduction

We will see some examples of linear programs soon.

2.2 Notation and Definitions

2.2.1 Useful Concepts

A *half-space* is a subset of $\mathbb{R}^n$ specified by a linear constraint as follows: $\{x \mid a^\intercal x \leq b\}$ for some $a \in \mathbb{R}^n$ and $b \in \mathbb{R}$. The intersection of a finite number of half-spaces is a *polyhedron*. A polyhedron K may be *bounded* (i.e., there exists some $M \in \mathbb{R}_{\geq 0}$ so that K is contained in the box $\{x \in \mathbb{R}^n \mid -M \leq x_i \leq M \forall i\}$) or *unbounded*. A bounded polyhedron is called a *polytope.*

Given a polyhedron K, a point $x \in K$ is a called a *vertex* or *extreme-point* or *corner* if there is some hyperplane $H := \{y \in \mathbb{R}^n \mid a^\intercal y = b\}$ such that $H \cap K = \{x\}$; that is, the point x is the unique point in the intersection of body and the hyperplane.

A set K is called *convex* if for any pair of points $x, y \in K$, all the points in the set $L := \{\alpha x + (1-\alpha)y \mid \alpha \in [0,1]\}$ also lie in K. It can be shown that the intersection of convex sets is convex. And that a half-space is convex. So a polyhedron is a convex set.

The polyhedron defined by the constraints of a linear program are called its *feasible region*. (It may be empty, in case the LP is infeasible.) The goal of linear programming is therefore to optimize a linear function over a polyhedron.

2.2.2 Notation for Linear Programs

Let us fix an LP that looks like the following:

For two vectors $c, x \in \mathbb{R}^n$, we will use $c^\intercal x$ to denote the inner product

$$\sum_i c_i x_i.$$

Other notation for the same concept include $c \cdot x$ and $\langle c, x \rangle$.

$$\begin{aligned} \max\ & c^\intercal x \\ Ax &\leq b \\ x &\geq 0. \end{aligned} \tag{2.1}$$

Let us assume that $x \in \mathbb{R}^n$ and the constraint matrix $A \in \mathbb{R}^{m \times n}$, so that there are m constraints (apart from the non-negativity ones).

1. We say that the LP is *infeasible* if there are no $x \in \mathbb{R}^n$ that satisfy all the constraints. E.g., the LP:

$$\begin{aligned} \max\ & x_1 \\ x_1 &\geq 2 \\ x_1 &\leq 1. \end{aligned}$$

2. We say that the LP is *unbounded* if there is no finite bound on the objective function. (For a maximization LP, we means that for every $N \in \mathbb{R}$, there is some $x \in \mathbb{R}^n$ such that $c^\intercal x \geq N$.) E.g., the LP:

$$\begin{aligned} \max\ & x_1 \\ x_1 &\geq 2. \end{aligned}$$

3. If the LP is not infeasible, and it is not unbounded, it means that the LP has *feasible* solutions x (ones that satisfy all the constraints), and that the optimal objective value is *bounded*. In this case we say that a solution x^* is *optimal* for the (maximization) LP if for all x that is feasible for the constraints, we have

$$c^\intercal x \leq c^\intercal x^*.$$

2.2.3 *The Main Algorithmic Result for LPs*

Theorem 2.1 (LPs solvable in poly-time)**.** *There are algorithms that given an LP, can correctly output whether it is* `infeasible` *or* `unbounded`*, or else output an optimal solution x^*, in time polynomial in the length of the input.*

Say all the numbers in the LP are integers of magnitude at most 2^B, then if A has m constraints we need $O(mnB)$ bits to write down the entire LP (2.1).

Recall that even though the LP may only have integer values in the objective and constraints, the optimal solution may have fractional coordinates. E.g., by inspection you can check that

$$\begin{aligned} \max\ & x_1 \\ x_1 + x_2 &\leq 1 \\ x_1 - x_2 &\leq 0 \end{aligned}$$

has optimal value 1/2 (and this is the only optimal solution).

Contrast this with the fact that for maximum flows—which are a special kind of LP—if the input contains all integer capacities, then there always exists an optimal max-flow that takes on integer values.

In fact, if we want to solve linear programs but restrict ourselves to integer solutions, then the problem becomes NP-hard.

Remark 2.2. Since the algorithm has to write down an optimal solution, Theorem 2.1 implies that the number of bits required to write down an optimal solution is also at most poly(mnB). This is not difficult to show: if you are intersted, check out the Matousek and Gärtner book.

2.3 *Duality*

We now discuss one of the central concepts of convex optimizatino: that of *duality*. Let us consider a linear program (which we call the *primal*):

$$\max c^\intercal x \tag{2.2}$$

$$Ax \leq b \tag{2.3}$$

$$x \geq 0. \tag{2.4}$$

We claim the following *dual* LP is an upper bound on the value of the primal LP:

See the 15-451 notes for now we derived this LP from first principles. Once you know how we came up with this LP, you don't have to derive it from first principles each time, but just use this syntactic approach to get duals.

$$\min b^\intercal y \tag{2.5}$$

$$A^\intercal y \geq c \tag{2.6}$$

$$y \geq 0.$$

Notice that we have a minimization problem in the dual instead of maximization in the primal: the roles of the objective function and right-hand side have been swapped, the constraint matrix has been transposed. (But the non-negativity constraints remain the same.)

Note that if the primal has n variables and m constraints, then the dual has m variables and n constraints.

Lemma 2.3 (Weak Duality Lemma)**.** *For any solution $x \in \mathbb{R}^n$ for the primal and any solution $y \in \mathbb{R}^m$ for the dual, we have*

$$c^\intercal x \leq b^\intercal y.$$

The assumption that both primal and dual programs have solutions means that both are feasible.

Proof. Since we know that $A^\intercal y \geq c$ because of the dual constraints (2.6), and that $x \geq 0$, we get

$$c^\intercal x \leq (A^\intercal y)^\intercal x = y^\intercal Ax.$$

But the primal constraints (2.3) tell us that $Ax \leq b$, and also $y \geq 0$, so

$$y^\intercal Ax \leq y^\intercal b = b^\intercal y. \qquad \square$$

The proof of weak duality was easy: a more sophisticated claim is the following:

Theorem 2.4 (Strong Duality Theorem)**.** *Suppose the primal and dual are both feasible. Then both primal and dual programs are bounded, and moreover they have the same optimal values*

$$\max_x c^\intercal x = \min_y b^\intercal y.$$

While weak duality arises in most convex optimization, strong duality is rarer for non-linear problems. But for linear optimization, strong duality is always true.

2.3.1 *Why Study Duality?*

Let us give some reasons to study this concept:

1. Duality gives us alternate ways to write the same problem. Suppose we want to solve a problem and we formulate it as the optimal value of an LP. Then we know that the dual of this LP also gives the same value, and hence gives another way of modeling the same problem.

   As an example, consider the problem of computing a shortest s-t path: one way was to write it as a minimum cost way to send one unit of flow from s to t in a network:

$$\begin{aligned} \min \quad & \sum_e c_e x_e \\ \sum_{e:e \text{ entering } v} x_e &= \sum_{e:e \text{ leaving } v} x_e \quad \forall v \notin \{s,t\} \\ \sum_{e:e \text{ leaving } s} x_e &= 1 \\ x_e &\geq 0 \quad \forall e. \end{aligned}$$

   If we take its dual (and massage it a bit) we get the following LP:

$$\begin{aligned} \max \quad & d_t \\ d_s &= 0 \\ d_v &\leq d_u + c_{uv} \quad \forall e = (u,v) \in E. \end{aligned}$$

   Sometimes it's easier to solve the primal problem, sometimes it's easier to solve the dual. (It all depends on the particular setting, so once you know LP duality, if you are worried about solving an LP, you can consider whether solving the dual is easier.)

You may see a problem using this idea in the HWs.

2. Duality allows you to easily certify that some solution x^* is an optimal solution to a (feasible and bounded) LP. Indeed, you can give a solution y^* to the dual with the same objective function value. (Such a y^* exists by Theorem 2.4.)

   By weak duality Lemma 2.3 we know that for every x feasible for the primal, we have

$$c^\intercal x \leq b^\intercal y^*.$$

   But that latter quantity equals $c^\intercal x^*$, so we get that

$$c^\intercal x \leq c^\intercal x^*$$

which shows that x^* is optimal.

We saw this in action when we discussed maxflow/mincut, since we could show that a flow was a maximum-value flow just by giving a matching minimum cut value.

3. Duality also arises in the workings of some algorithms to solve LPs (e.g., interior point algorithms), even though you don't need to deal with these algorithms directly.

4. And finally, linear programming duality leads naturally into convex duality, which we may discuss later.

2.4 *Algorithms to Solve Linear Programs*

There are many different algorithms that can be used to solve linear programs. The main ones are the *Simplex*, the *Ellipsoid*, and *Interior Point* algorithms.

2.4.1 *The Simplex Algorithm*

Since the