14. Linear Programming

Lecture on Thursday 12^{th} November, 2009 by Bernd Gärtner <gaertner@inf.ethz.ch>

This lecture is about a special type of optimization problems, namely *linear programs*. We start with a geometric problem that can directly be formulated as a linear program.

14.1 Linear Separability of Point Sets

Let $P \subseteq \mathbb{R}^d$ and $Q \subseteq \mathbb{R}^d$ be two finite point sets in d-dimensional space. We want to know whether there exists a hyperplane that separates P from Q (we allow non-strict separation, i.e. some points are allowed to be on the hyperplane). Figure 14.1 illustrates the 2-dimensional case.

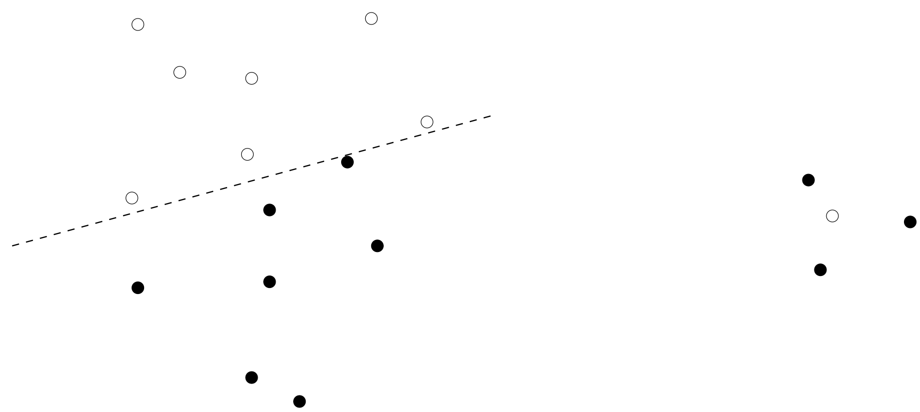


Figure 14.1: *Left: there is a separating hyperplane; Right: there is no separating hyperplane*

How can we formalize this problem? A hyperplane is a set of the form

$$h = \{x \in \mathbb{R}^d : h_1x_1 + h_2x_2 + \cdots + h_dx_d = h_0\},$$

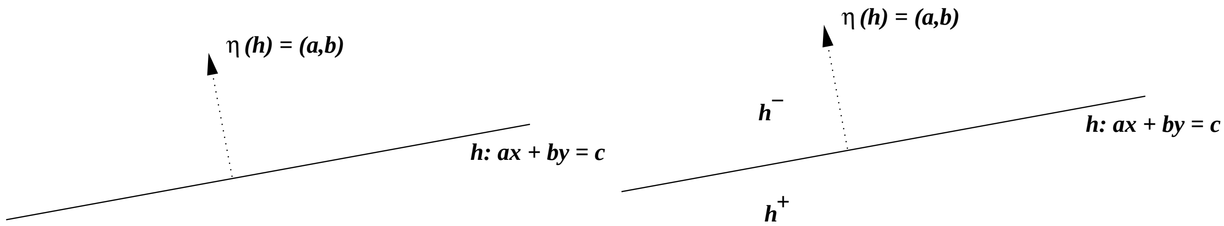
where $h_i \in \mathbb{R}, i = 0, \ldots, d$. For example, a line in the plane has an equation of the form $ax + by = c$.

The vector $\eta(h) = (h_1, h_2, \ldots, h_d) \in \mathbb{R}^d$ is called the *normal vector* of h. It is orthogonal to the hyperplane and usually visualized as in Figure 14.2(a).

Every hyperplane h defines two closed *halfspaces*

$$\begin{aligned} h^+ &= \{x \in \mathbb{R}^d : h_1x_1 + h_2x_2 + \cdots + h_dx_d \leq h_0\}, \\ h^- &= \{x \in \mathbb{R}^d : h_1x_1 + h_2x_2 + \cdots + h_dx_d \geq h_0\}. \end{aligned}$$

Each of the two halfpsaces is the region of space "on one side" of h (including h itself). The normal vector $\eta(h)$ points into h^-, see Figure 14.2(b). Now we can formally define linear separability.



(a) The normal vector of a hyperplane (b) The two halfspaces of a hyperplane

Figure 14.2: *The concepts of hyperplane, normal vector, and halfspace*

Definition 14.1 *Two point sets* $P \subseteq \mathbb{R}^d$ *and* $Q \subseteq \mathbb{R}^d$ *are called* linearly separable *if there exists a hyperplane* h *such that* $P \subseteq h^+$ *and* $Q \subseteq h^-$. *In formulas, if there exist real numbers* $h_0, h_1, \ldots, h_d$ *such that*

$$\begin{aligned} h_1p_1 + h_2p_2 + \cdots + h_dp_d &\leq h_0, \quad p \in P, \\ h_1q_1 + h_2q_2 + \cdots + h_dq_d &\geq h_0, \quad q \in Q. \end{aligned}$$

As we see from Figure 14.1, such $h_0, h_1, \ldots, h_d$ may or may not exist. How can we find out?

14.2 Linear Programming

The problem of testing for linear separability of point sets is a special case of the general linear programming problem.

Definition 14.2 *Given* $n, d \in \mathbb{N}$ *and real numers*

$$\begin{aligned} b_i \quad &, \quad i = 1, \ldots, n, \\ c_j \quad &, \quad j = 1, \ldots, d, \\ a_{ij} \quad &, \quad i = 1, \ldots, n, \quad j = 1, \ldots, d, \end{aligned}$$

the linear program *defined by these numbers is the problem of finding real numbers* $x_1, x_2, \ldots, x_d$ *such that*

(i) $\sum_{j=1}^{d} a_{ij}x_j \leq b_i, \quad i = 1, \ldots, n,$ *and*

(ii) $\sum_{j=1}^{d} c_jx_j$ *is as large as possible subject to (i).*

Let us get a geometric intuition: each of the n *constraints* in (i) requires $x = (x_1, x_2, \ldots, x_d) \in \mathbb{R}^d$ to be in the positive halfspace of some hyperplane. The intersection of all these halfspaces is the *feasible region* of the linear program. If it is empty, there is no solution—the linear program is called *infeasible*.

Otherwise—and now (ii) enters the picture—we are looking for a *feasible solution* x (a point inside the feasible region) that maximizes $\sum_{j=1}^{d} c_j x_j$. For every possible value γ of this sum, the feasible solutions for which the sum attains this value are contained in the hyperplane

$$\{x \in \mathbb{R}^d : \sum_{j=1}^{d} c_j x_j = \gamma\}$$

with normal vector $c = (c_1, \ldots, c_d)$. Increasing γ means to shift the hyperplane into direction c. The highest γ is thus obtained from the hyperplane that is most extreme in direction c among all hyperplanes that intersect the feasible region, see Figure 14.3.

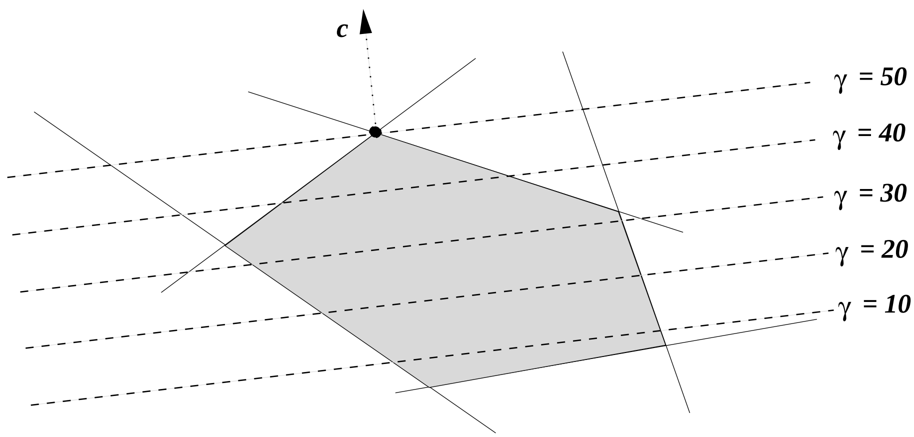


Figure 14.3: *A linear program: finding the feasible solution in the intersection of five positive halfspaces that is most extreme in direction* c *(has highest value* $\gamma = \sum_{j=1}^{d} c_j x_j$*)*

In Figure 14.3, we do have an *optimal solution* (a feasible solution x of highest value $\sum_{j=1}^{d} c_j x_j$), but in general, there might be feasible solutions of arbitrarily high γ-value. In this case, the linear program is called *unbounded*, see Figure 14.4.

It can be shown that infeasibility and unboundedness are the only obstacles for the existence of an optimal solution. If the linear program is feasible and bounded, there exists an optimal solution.

This is not entirely trivial, though. To appreciate the statement, consider the problem of finding a point (x, y) that (i) satisfies $y \geq e^x$ and (ii) has smallest value of y among all (x, y) that satisfy (i). This is not a linear program, but in the above sense it is feasible (there are (x, y) that satisfy (i)) and bounded (y is bounded below from 0 over the set of feasible solutions). Still, there is no optimal solution, since values of y arbitrarily close to 0 can be attained but not 0 itself.

Even if a linear program has an optimal solution, it is in general not unique. For example, if you rotate c in Figure 14.3 such that it becomes orthogonal to the top-right

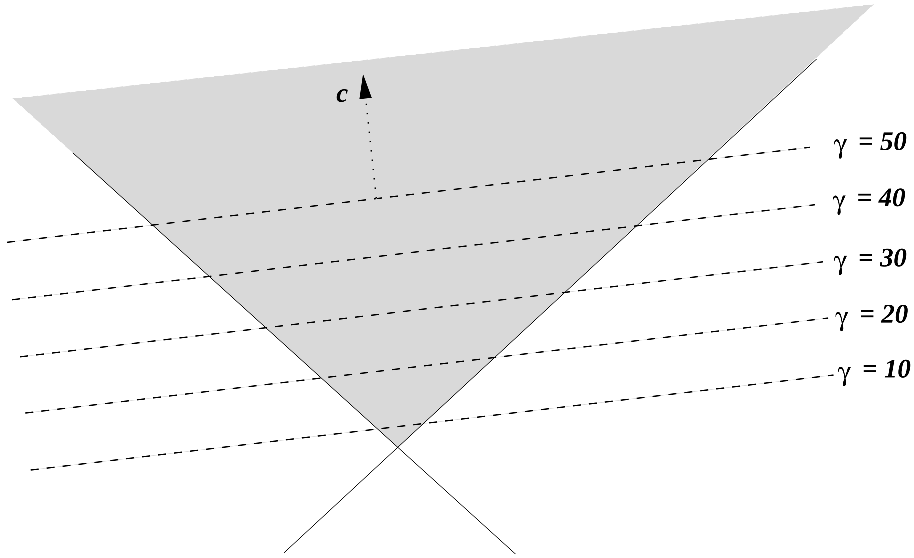


Figure 14.4: *An unbounded linear program*

edge of the feasible region, then every point of this edge is an optimal solution. Why is this called a *linear* program? Because all constraints are linear inequalities, and the objective function is a linear function. There is also a reason why it is called a linear *program*, but we won't get into this here (see [3] for more background).

Using vector and matrix notation, a linear program can succinctly be written as follows.

$$\text{(LP)} \quad \begin{array}{ll} \text{maximize} & c^T x \\ \text{subject to} & Ax \le b \end{array}$$

Here, $c, x \in \mathbb{R}^d, b \in \mathbb{R}^n, A \in \mathbb{R}^{n \times d}$, and $\cdot^T$ denotes the transpose operation. The inequality "$\le$" is interpreted componentwise. The vector x represents the *variables*, c is called the *objective function vector*, b the *right-hand side*, and A the *constraint matrix*.

To *solve* a linear programs means to either report that the problem is infeasible or unbounded, or to compute an optimal solution x^*. If we can solve linear programs, we can also decide linear separability of point sets. For this, we check whether the linear program

$$\begin{array}{lll} \text{maximize} & 0 & \\ \text{subject to} & h_1 p_1 + h_2 p_2 + \cdots + h_d p_d - h_0 \le 0, & p \in P, \\ & h_1 q_1 + h_2 q_2 + \cdots + h_d q_d - h_0 \ge 0, & q \in Q. \end{array}$$

in the $d+1$ variables $h_0, h_1, h_2, \ldots, h_d$ and objective function vector $c = 0$ is feasible or not. The fact that some inequalities are of the form "$\geq$" is no problem, of course, since we can multiply an inequality by -1 to turn "$\geq$" into "$\leq$".

14.3 Minimum-area Enclosing Annulus

Here is another geometric problem that we can write as a linear program, although this is less obvious. Given a point set $P \subseteq \mathbb{R}^2$, find a *minimum-area annulus* (region between two concentric circles) that contains P; see Figure 14.5 for an illustration.

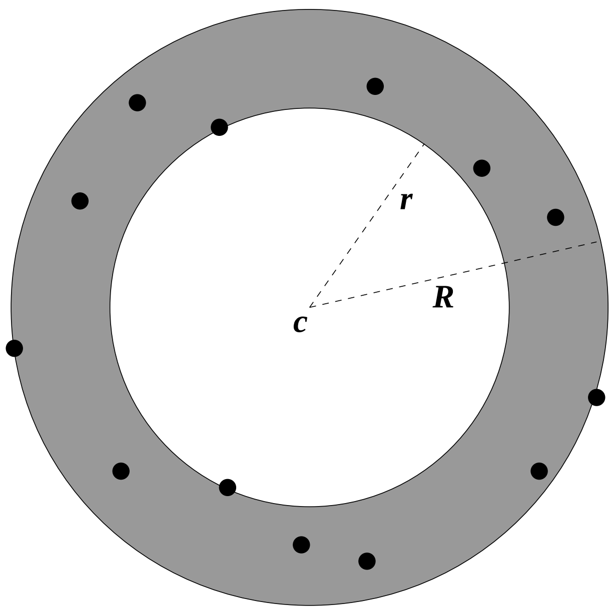


Figure 14.5: *A minimum-area annulus containing* P

The optimal annulus can be used to test whether the point set P is (approximately) on a common circle which is the case if the annulus has zero (or small) area.

Let us write this as an optimization problem in the variables $c = (c_1, c_2) \in \mathbb{R}^2$ (the center) and $r, R \in \mathbb{R}$ (the small and the large radius).

$$\begin{array}{ll} \text{minimize} & \pi(R^2 - r^2) \\ \text{subject to} & r \leq \|p - c\| \leq R, \quad p \in P. \end{array}$$

This neither has a linear objective function nor are the constraints linear inequalities. But a variable substitution will take care of this. We define new variables

$$u := r^2 - \|c\|^2, \tag{14.3}$$

$$v := R^2 - \|c\|^2. \tag{14.4}$$

Omitting the factor π in the objective function does not affect the optimal solution (only its value), hence we can equivalently work with the objective function $v - u = R^2 - r^2$. The constraint $r \leq ||p - c||$ is equivalent to $r^2 \leq ||p||^2 - 2p^T c + ||c||^2$, or

$$u + 2p^T c \leq ||p||^2.$$

In the same way, $||p - c|| \leq R$ turns out to be equivalent to

$$v + 2p^T c \geq ||p||^2.$$

This means, we now have a *linear* program in the variables u, v, c_1, c_2:

$$\begin{array}{ll} \text{minimize} & v - u \\ \text{subject to} & u + 2p^T c \leq ||p||^2, \quad p \in P \\ & v + 2p^T c \geq ||p||^2, \quad p \in P. \end{array}$$

From optimal values for u, v and c, we can also reconstruct r and R^2 via (14.3) and (14.4).

14.4 Solving a Linear Program

Linear programming was first studied in the 1930's -1950's, and some of its original applications were of a military nature. In the 1950's, Dantzig invented the *simplex method* for solving linear programs, a method that is fast in practice but is not known to come with any theoretical guarantees [1].

The computational complexity of solving linear programs was unresolved until 1979 when Leonid Khachiyan discovered a polynomial-time algorithm known as the *ellipsoid method* [2]. This result even made it into the New York times.

From a computational geometry point of view, linear programs with a *fixed* number of variables are of particular interest (see our two applications above, with d and 4 variables, respectively, where d may be 2 or 3 in some relavant cases). As was first shown by Megiddo, such linear programs can be solved in time $O(n)$, where n is the number of constraints [4]. In the next lecture, we will describe a much simpler randomized $O(n)$ algorithm due to Seidel [5].

Questions

42. *What is a linear program?* Give a precise definition! How can you visualize a linear program? What does it mean that the linear program is infeasible / unbounded?

43. *Show an application of linear programming!* Describe a geometric problem that can be formulated as a linear program, and give that formulation!

References

[1] G. B. Dantzig, *Linear Programming and Extensions*, Princeton University Press, Princeton, NJ, 1963.

[2] L. G. Khachiyan, Polynomial algorithms in linear programming, *U.S.S.R. Comput. Math. and Math. Phys* **20** (1980), 53–72.

[3] J. Matoušek and B. Gärtner, *Understanding and Using Linear Programming*, Universitext, Springer-Verlag, 2007.

[4] N. Megiddo, Linear programming in linear time when the dimension is fixed, *J. ACM* **31** (1984), 114–127.

[5] R. Seidel, Small-dimensional linear programming and convex hulls made easy, *Discrete Comput. Geom.* **6** (1991), 423–434.