5.1 Linear Programming

5.1.1 Definition and standard forms

Definition. A linear optimization problem (or, linear program, LP) is one of the standard form:

\[
\min_x f_0(x) : f_i(x) \leq 0, \quad i = 1, \ldots, m, \\
Ax = b, \quad i = 1, \ldots, p,
\]

where every function \( f_0, f_1, \ldots, f_m \) is affine. Thus, the feasible set of an LP is a polyhedron.

Standard forms. Linear optimization problems admits several standard forms. One is derived from the general standard form:

\[
\min_x c^T x + d : Ax = b, \quad Cx \leq h,
\]

where the inequalities are understood componentwise\(^1\). The constant term \( d \) in the objective function is, of course, immaterial.

Another standard form—used in several off-the-shelf algorithms—is

\[
\min_x c^T x : Ax = b, \quad x \geq 0. \tag{5.1}
\]

We can always transform the above problem into the previous standard form, and vice-versa.

5.1.2 Examples

Piece-wise linear minimization. A piece-wise linear function is the point-wise maximum of affine functions, and has the form

\[
f(x) = \max_{1 \leq i \leq m} (a_i^T x + b_i),
\]

for appropriate vectors \( a_i \) and scalars \( b_i, i = 1, \ldots, m \). The (unconstrained) problem of minimizing the piece-wise linear function above is not an LP. However, its epigraph form:

\[
\min_{x,t} t : t \geq a_i^T x + b_i, \quad i = 1, \ldots, m
\]

is.

\(^1\)Notice that the convention for componentwise inequalities differs from the one adopted in BV. I will reserve the symbol \( \preceq \) or \( \succeq \) for negative and positive semi-definiteness of symmetric matrices.
Figure 5.1. A linear classifier, with a total of four errors on the training set. The sum of the lengths of the dotted lines (which correspond to classification errors) is the value of the loss function at optimum.

$l_1$-norm regression. A related example involves the minimization of the $l_1$-norm of a vector that depends affinely on the variable. This arises in regression problems, such as image reconstruction. Using a notation similar to the previous example, the problem has the form

$$\min_x \sum_{i=1}^m |a_i^T x + b_i|.$$ 

The problem is not an LP, but we can introduce slack variables and re-write the above in the equivalent, LP format:

$$\min_{x,v} \sum_{i=1}^m v_i : -v_i \leq a_i^T x + b_i \leq v_i, \ i = 1, \ldots, m.$$ 

Linear binary classification. Consider a two-class classification problem as shown in Figure 5.1. Given $m$ data points $x_i \in \mathbb{R}^n$, each of which is associated with a label $y_i \in \{-1, 1\}$, the problem is to find a hyperplane that separates, as much as possible, the two classes.

The two classes are separable by a hyperplane $\mathcal{H}(w,b) = \{x : w^T x + b \leq 0\}$, where $w \in \mathbb{R}^n$, $w \neq 0$, and $b \in \mathbb{R}$, if and only if $w^T x_i + b \geq 0$ for $y_i = +1$, and $w^T x_i + b \leq 0$ if $y_i = -1$. Thus, the conditions on $(w,b)$

$$y_i (w^T x_i + b) \geq 0, \ i = 1, \ldots, m$$

ensure that the data set is separable by a linear classifier. In this case, the parameters $w, b$ allow to predict the label associated to a new point $x$, via $y = \text{sign}(w^T x + b)$. The feasibility problem—finding $(w, b)$ that satisfy the above separability constraints—is an LP. If the data
set is strictly separable (every condition in (5.2) holds strictly), then we can re-scale the constraints and transform them into

$$y_i(w^T x_i + b) \geq 1, \quad i = 1, \ldots, m.$$  \hspace{1cm} (5.2)

In practice, the two classes may not be linearly separable. In this case, we would like to minimize, by proper choice of the hyperplane parameters, the total number of classification errors. Our objective function has the form

$$\sum_{i=1}^{m} \psi(y_i(w^T x_i + b)),$$

where $\psi(t) = 1$ if $t < 0$, and 0 otherwise.

Unfortunately, the above objective function is not convex, and hard to minimize. We can replace it by an upper bound, which is called the hinge function, $h(t) = (1 - t)_+ = \max(0, 1 - t)$. Our problem becomes one of minimizing a piece-wise linear “loss” function:

$$\min_{w, b} \sum_{i=1}^{m} (1 - y_i(w^T x_i + b))_+.$$

In the above form, the problem is not yet an LP. We may introduce slack variables to obtain the LP form:

$$\min_{w, b, v} \sum_{i=1}^{m} v_i : \quad v \geq 0, \quad y_i(w^T x_i + b) \geq 1 - v_i, \quad i = 1, \ldots, m.$$

The above can be seen as a variant of the separability conditions (5.2), where we allow for infeasibilities, and seek to minimize their sum. The value of the loss function at optimum can be read from Figure 5.1: it is the sum of the lengths of the dotted lines, from data points that are wrongly classified, to the hyperplane.

Network flow. Consider a network (directed graph) having $m$ nodes connected by $n$ directed arcs (ordered pairs $(i, j)$). We assume there is at most one arc from node $i$ to node $j$, and no self-loops. We define the arc-node incidence matrix $A \in \mathbb{R}^{m \times n}$ to be the matrix with coefficients $A_{ij} = 1$ if arc $j$ starts at node $i$, $-1$ if it ends there, and 0 otherwise. Note that the column sums of $A$ are zero: $1^T A = 0$.

A flow (of traffic, information, charge) is represented by a vector $x \in \mathbb{R}^n$, and the total flow leaving node $i$ is then $(Ax)_i = \sum_{j=1}^{n} A_{ij} x_j$.

The minimum cost network flow problem has the LP form

$$\min_x c^T x : \quad Ax = b, \quad l \leq x \leq u,$$

where $c_i$ is the cost of flow through arc $i$, $l, u$ provide upper and lower bounds on $x$ and $b \in \mathbb{R}^m$ is an external supply vector. This vector may have positive or negative components,
as it represents supply and demand. We assume that $1^T b = 0$, so that the total supply equals the total demand. The constraint $Ax = b$ represents the balance equations of the network.

A more specific example is the max flow problem, where we seek to maximize the flow between node 1 (the source) and node $m$ (the sink). It bears the form

$$\min_{x,t} \; t : \; Ax = te, \; l \leq x \leq u,$$

with $e = (1, 0, \ldots, 0, -1)$.

**LP relaxation of boolean problems.** A boolean optimization problem is one where the variables are constrained to be boolean. An example of boolean problem is the so-called boolean LP

$$p^* = \min_x \; c^T x : \; Ax \leq b, \; x \in \{0, 1\}^n.$$ 

Such problems are non-convex, and usually hard to solve. The LP relaxation takes the form

$$p^*_{LP} := \min_x \; c^T x : \; Ax \leq b, \; 0 \leq x \leq 1.$$ 

The relaxation provides a lower bound on the original problem: $p^*_{LP} \leq p^*$. Hence, its optimal points may not be feasible (not boolean). Even though a solution of the LP relaxation may not necessarily be boolean, we can often interpret it as a fractional solution to the original problem. For example, in a graph coloring problem, the LP relaxation colors the nodes of the graph not with a single color, but with many.

Boolean problems are not always hard to solve. Indeed, in some cases, one can show that the LP relaxation provides an exact solution to the boolean problem, as optimal points turn out to be boolean. A few examples in this category, involving network flow problems with boolean variables:

- The **weighted bipartite matching** problem is to match $N$ people to $N$ tasks, in a one-to-one fashion. The cost of matching person $i$ to task $j$ is $A_{ij}$. The problem reads

$$\min_x \sum_{i,j=1}^N A_{ij} x_{ij} : \forall j, \sum_{i=1}^N x_{ij} = 1 \quad (\text{one person for each task}) \quad \forall i, \sum_{j=1}^N x_{ij} = 1 \quad (\text{one task for each person})$$

- The **shortest path** problem has the form

$$\min_x \; 1^T x : \; Ax = (1, 0, \ldots, 0, -1), \; x \in \{0, 1\}^n,$$

where $A$ stands for the incidence matrix of the network, and arcs with $x_i = 1$ form a shortest forward path between nodes 1 and $m$. As before the LP relaxation in this case is exact, in the sense that its solution is boolean. (The LP relaxation problem can be solved very efficiently with specialized algorithms.)
5.2 Overview of conic optimization

Conic optimization models. The linear optimization model can be written in standard form as
\[
\min_{x} c^T x : Ax = b, \ x \geq 0,
\]
where we express the feasible set as the intersection of an affine subspace \( \{x : Ax = b\} \), with the non-negative orthant, \( \mathbb{R}^n_+ \). One can think of the linear equality constraints, and the objective, as the part in the problem that involves the data \((A, b, c)\), while the sign constraints describe its structure.

With the advent of provably polynomial-time methods for linear optimization in the late 70’s, researchers tried to generalize the linear optimization model, in a way that retained the nice complexity properties of the linear model.

Early attempts at generalizing the above model focussed on allowing the linear map \( x \rightarrow Ax \) to be nonlinear. Unfortunately, as soon as we introduce non-linearities in the equality constraints, the model becomes non-convex and potentially intractable numerically. Thus, modifying the linear equality constraints is probably not the right way to go.

Instead, one can try to modify the "structural" constraints \( x \in \mathbb{R}^n_+ \). If one replaces the non-negative orthant with another set \( \mathcal{K} \), then we obtain a generalization of linear optimization. Clearly, we need \( \mathcal{K} \) to be a convex set, and we can further assume it to be a cone (if not, we can always introduce a new variable and a new equality constraint in order to satisfy this condition). Hence the motivation for the so-called conic optimization model:
\[
\min_{x} c^T x : Ax = b, \ x \in \mathcal{K},
\]
where \( \mathcal{K} \) is a given convex cone.

The issue becomes then of finding those convex cones \( \mathcal{K} \) for which one can adapt the efficient methods invented for linear optimization, to the conic problem above. A nice theory due to Nesterov and Nemirovski, which they developed in the late 80’s, allows to find a rich class of cones for which the corresponding conic optimization problem is numerically tractable. We refer to this class as tractable conic optimization.

Tractable conic optimization. The cones that are “allowed” in tractable conic optimization are of three basic types, and include any combination (as detailed below) of these three basic types. The three basic cones are

- The non-negative orthant, \( \mathbb{R}^n_+ \). (Hence, conic optimization includes linear optimization as a special case.)
- The second-order cone, \( \mathcal{Q}^n := \{(x, t) \in \mathbb{R}^n_+ : t \geq \|x\|_2\} \).
- The semi-definite cone, \( \mathcal{S}^n_+ = \{X = X^T \succeq 0\} \).
A variation on the second-order cone, which is useful in applications, involves the rotated second-order cone $Q_n^{\text{rot}} := \{(x, y, z) \in \mathbb{R}^{n+2} : 2yz \geq \|x\|_2^2, y \geq 0, z \geq 0\}$. We can easily convert the rotated second-order cone into the ordinary second-order cone representation, since the constraints $2yz \geq \|x\|_2^2, y \geq 0, z \geq 0$, are equivalent to

$$(y + z) \geq \frac{(y - z)}{\sqrt{2x}}_2.$$ 

We can build all sorts of cones that are admissible for the tractable conic optimization model, using combinations of these cones. For example, in a specific instance of the problem, we might have constraints of the form

$$x_1 \geq 0, \ x_3 \geq \sqrt{x_1^2 + x_2^2}, \ \left(\begin{array}{ccc} x_2 & x_4 \\ x_4 & x_5 \end{array}\right) \succeq 0.$$ 

The above set of constraints involves the non-negative orthant (first constraint), the second-order cone (second constraint), and the third, the semi-definite cone.

We can always introduce new variables and equality constraints, to make sure that the cone $K$ is a direct product of the form $K_1 \times \ldots \times K_m$, where each $K_i$ is a cone of one of the three basic types above. In the example above, since the variable $x_2$ appears in two of the cones, we add a new variable $x_6$ and the equality constraint $x_2 = x_6$. With that constraint, the constraint above can be written $x = (x_1, \ldots, x_6) \in K$, where $K$ is the direct product $\mathbb{R}_+ \times Q^2 \times S^2_+$. 

Note that the three basic cones are nested, in the sense that we can interpret the non-negative orthant as the projection of a direct product of second-order cones on a subspace (think of imposing $x = 0$ in the definition of $Q^n$). Likewise, a projection of the semi-definite cone on a specific subspace gives the second-order cone, since

$$\|x\|_2 \leq t \iff \begin{pmatrix} t & x_1 & \ldots & x_n \\ x_1 & t & \ & 0 \\ \vdots & \ & \ddots & \ \\ x_n & 0 & \ldots & t \end{pmatrix} \succeq 0. \quad (5.4)$$ 

(The proof of this exercise hinges on the Schur complement lemma, see BV, pages 650-651.)

**Exercises**

1. Prove that any linear program can be expressed in the standard form 5.1. Justify your steps carefully.

2. Prove the result (5.4).