10-725: Optimization **Fall 2012**

Lecture 13: October 9

Lecturer: Geoff Gordon *Scribes: Carl Doersch, Sashank Reddi*

Note: *LaTeX template courtesy of UC Berkeley EECS dept.*

13.1 Linear Programs (and Fisher Scoring)

13.1.1 Fisher Scoring

Fisher's scoring algorithm is related to Newton's method from last time. Recall that, for an exponential family parameterized by θ, the Newton update to maximize the likelihood has a very simple form:

$$Var[X|\theta]d\theta = E[x|\theta] - \bar{x}$$

I.e. the Hessian is simply the variance. While this is exact for exponential families, it's often a good approximation for arbitrary distributions as well. It has some nice properties, for example:

- The direction chosen is guaranteed to be a descent direction, since the Hessian is positive semidefinite.
- The variance is independent of the data, and often has a simple, known form.
- It often has a wider radius of convergence than Newton's. (though there are no theoretical results about why, it just seems to happen often in practice)
- It can be superlinearly convergent.

13.2 Linear Programs

Thus far we've focused on optimizing arbitrary convex functions, but in linear programs, the we are optimizing a simple linear function. The added challenge of the linear program (LP), however, is the constraint set.

13.2.1 The Setup

The setup is as follows:

A set of variables:

$$x = (x_1, ..., x_n)^\top$$

A linear objective function:

$$\min_x \left(\text{or} \max_x\right) c^\top x$$

And a set of constraints (note we may have ranges for the variables, but these can be trivially rewritten as inequality constraints):

$$a_j^\top x = (\text{or} \leq \text{or} \geq)\, b_j \quad \text{for} \quad j = \{1, ..., m\}$$

For example:

$$\max_{x,y} \quad 2x + 3y \quad \text{s.t.} \tag{13.1}$$

$$\begin{aligned} x + y &\leq 4 \\ 2x + 5y &\leq 12 \\ x + 2y &\leq 5 \\ x, y &\geq 0 \end{aligned}$$

In 2d, it is straightforward to plot these constraints as half-spaces, as shown in Figure 13.1.

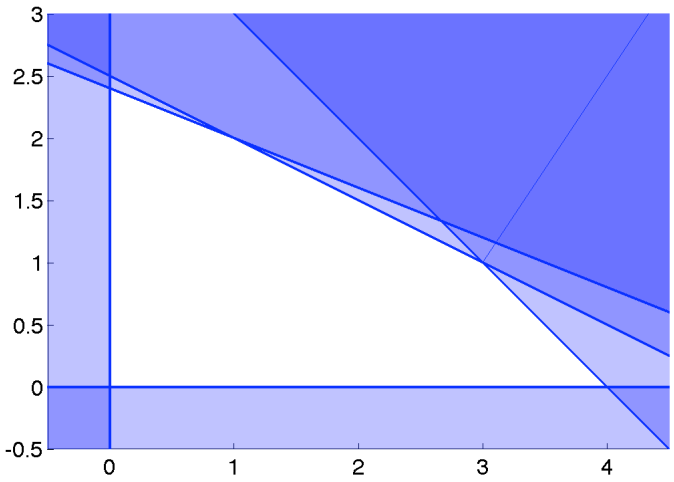


Figure 13.1: A plot of the constraints in the linear program in equation 13.2

A few definitions:

- The region where all constraints are satisfied (white in Figure 13.1) is the **feasible region**. and the region where at least one constraint is violated (blue in 13.1) is the **infeasible region**.
- With respect a point x in the feasible region, a constraint $a_j^\top x \leq (\text{or} \geq)\, b_j$ is **active** if $a_j^\top x = 0$; otherwise it is **inactive**.

- A **polyhedron** is defined as the intersection of half-spaces, but also arises as the convex hull of a set of points.

- Say a polyhedron lies in n-dimensional space, and we select a set of constraints Q (such that Q is not redundant, in the sense that none of the constraints can be removed without changing the feasible region). Let $d = n - |Q|$. Let S be the set of points in the feasible region of this polyhedron where all constraints in Q are active. If S is nonempty, then S is called a **d-face** of the polyhedron. A d-face is d-dimensional. Note that, if the polyhedron has nonzero volume, the n-face is the entire set.

- A 0-face is a single point called a **vertex**.

- A $n-1$-face is called a **facet**.

Note that the optimal value of the LP will always be achieved by at least one vertex, though there may be other optimal points as well. The set of points that achieve the optimum will comprise a d-face for some d.

It is generally more concise to write an LP in matrix format, i.e. to convert every constraint into a $\leq$ constraint (by negation, or by writing an equality as two inequalities), and writing:

$$\min_v \; c^\top v \quad \text{subj. to} \\ Av \leq b$$

Here, the $\leq$ is componentwise; i.e. row i of A and b_i form one constraint from the original LP. For example, we can rewrite 13.2 as:

$$A = \begin{pmatrix} 1 & 1 \\ 2 & 5 \\ 1 & 2 \\ -1 & 0 \\ 0 & -1 \end{pmatrix} \quad b = \begin{pmatrix} 4 \\ 12 \\ 5 \\ 0 \\ 0 \end{pmatrix} \quad c = \begin{pmatrix} 2 \\ 3 \end{pmatrix}$$

13.2.2 Solving LP's

In terms of finding the x that optimizes $c^\top x$, all that matters is the direction of the gradient of $c^\top x$. Thus, in 2-d, we could imagine rotating the space such that this gradient points down (if we're maximizing) or up (if we're minimizing). Then the optimum is simply the lowest point. We could imagine dropping a ball somewhere in the feasible region and letting it roll to the bottom. Applying this to equation 13.2 gives this:

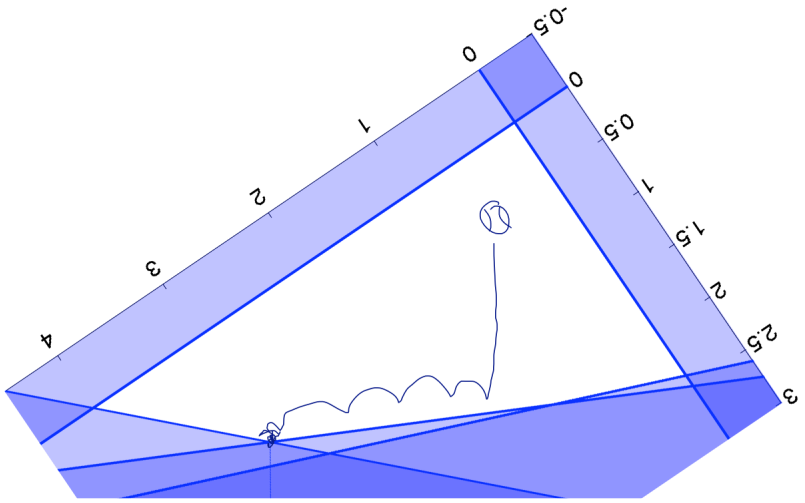


Figure 13.2: Rotated version of equation 13.2, and the ball-rolling solution to it.

However, this goes wrong in the case below, illustrating why gradient descent may not always be the best idea:

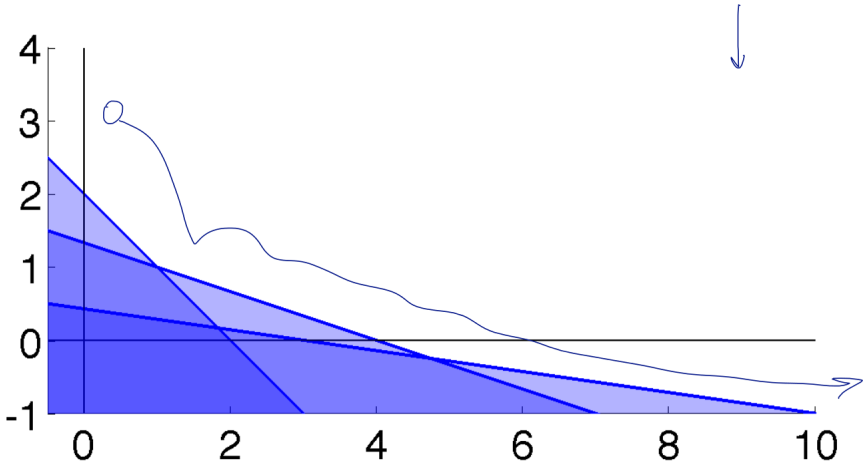


Figure 13.3: Where ball-rolling fails.

We can also encounter cases where the feasible region is empty. In this case, the convention is that the max over the empty set is $-\infty$ and the min is ∞. That way, adding a point to the set is guaranteed to update the current optimum value.

More practically, the first step of solving an LP is to find a feasible point. But how hard is it to find a feasible point? It turns out that it's essentially just as hard as solving the original LP. We can see this via

a reduction. Say we have an LP of the form: maximize $c^\top x$ subject to $Ax \leq b$. Say that we also have a solver that will tell us whether a feasible point exists. Then we can augment A and b with a constraint of the form $c^\top x \geq t$ for any t we wish, and the solver will let us know whether a feasible point exists, i.e., whether it is possible to make our objective $\geq t$. We can then perform a binary search on t, which will zero in on the largest value that $c^\top x$ can achieve. Hence we can solve the original LP with a number of calls to the feasibility solver that's linear in the number of bits of accuracy we desire.

13.2.3 Transforming LPs

Naïvely solving LPs is hard, but very often a hard LP can be transformed into an easier one. Here are four useful transformations:

1. We can rewrite the inequality $x+y \leq 4$ by replacing it with the constraint $x+y+s = 4, s \geq 0$. Hence, the inequality on multiple variables becomes a range constraint on a single variable.
2. An equality constraint like $x + 2y = 4$ can be replaced with two inequality constraints $x + 2y \leq 4$ and $x + 2y \geq 4$. (Note, however, that this transformation can cause problems with some optimization algorithms)
3. Say that we do not want to deal with negative values for variables (more on this below). A free variable x can be rewritten as the difference of two variables that are constrained to be positive, i.e. $x = a - b, a, b \geq 0$.
4. A bounded variable $x \in [a, b]$ can be rewritten as $x \geq a$, $x \leq b$.

Using (1), (3), and (4), we can rewrite LP's in **standard form**. When an LP is in standard form, all variables are constrained to be nonnegative, and all other constraints are equalities. In other words, the LP is written as $\max_q \; c^\top q$, subject to $Aq = b$, $q \geq 0$.

Returning to the LP in equation 13.2, we let $q = [x \;\; y \;\; u \;\; v \;\; w]$. Then the standard form is:

$$A = \begin{pmatrix} 1 & 1 & 1 & 0 & 0 \\ 2 & 5 & 0 & 1 & 0 \\ 1 & 2 & 0 & 0 & 1 \end{pmatrix} \quad b = \begin{pmatrix} 4 \\ 12 \\ 5 \end{pmatrix} \quad c = \begin{pmatrix} 2 \\ 3 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

By convention, this is often written in what is called a **tableau**:

x	y	u	v	w	z	RHS
1	1	1	0	0	0	4
2	5	0	1	0	0	12
1	2	0	0	1	0	5
-2	-3	0	0	0	1	0

Note the special variable z. Here, rather than optimizing the value of $c^\top x$, we simply define z such that $c^\top x = z$ and optimize z.

Note also that if there are n variables and $m \leq n$ equality constraints in the standard form of the LP, there must be n inequality constraints and $n - m$ variables in the inequality form. To see this, note that

conversion from the inequality form to the standard form requires replacing each inequality with exactly one slack variable, constrained to be above 0 (geometrically, the number of half-spaces constraining the LP cannot change). In Example.13.2, we can verify that m=3 and n=5, and therefore that there were 2 variables and 5 inequalities in the original. Standard form has a higher dimension than the inequality form. This is due to the introduction of the slack variables. Also note that setting the slack variables to 0 and finding the solution of standard form is effectively finding the corner points in the inequality form.

13.2.4 Faces in Inequality and Standard form

How do faces in Inequality and Standard form look? Recall that a d-face makes n-d inequality constraints tight. Inequality form with n variables and $m \geq n$ halfspaces, can have 0-face through n-faces.In contrast, standard form with n variables which are non-negative and $m \leq n$ equalities can have 0-face through n-m faces.

13.2.5 Why is the Standard form useful?

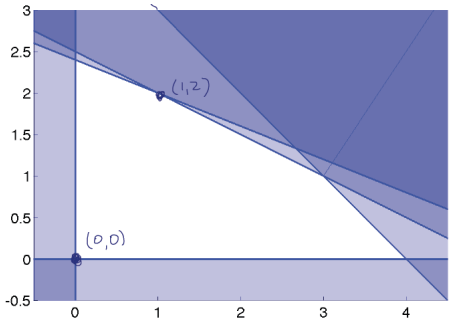
Standard form enables us to use tools from linear algebra to solve the LP. Equality constraints can be handled using gaussian elimination. We can easily transform the equality constraints in standard form using linear combinations and row operations to determine the corner points of the feasible region while these operations are much harder in the inequality form.

Example

Let us continue with Example 13.2 to explain more about these concepts.

x	y	u	v	w	z	
1	1	1	0	0	0	4
2	5	0	1	0	0	12
1	2	0	0	1	0	5
-2	-3	0	0	0	1	0

Recall that u, v, w are the slack variables. The final row represents the objective function. By setting $x, y = 0$, we can determine the value of the other variables i.e $u = 4, v = 12, w = 5$. This is lead to a corner of the feasible region (as shown in figure below) in inequality form. This can also be seen as projections on the subspace spanned the slack variables. Note that this operation may lead to infeasible solutions. For example, if we set $x, u = 0$, we get $y = 4, v = -8, w = -3$. This refers to the corner $(0, 4)$ in the inequality form, which lies in the infeasible region.



13.2.6 Row operations

We can replace any row with linear combination of existing rows as long as it does not loose independence. This will give us a general transformation in standard form . For example: we can eliminate x from 2nd and 3rd rows and replace it with:

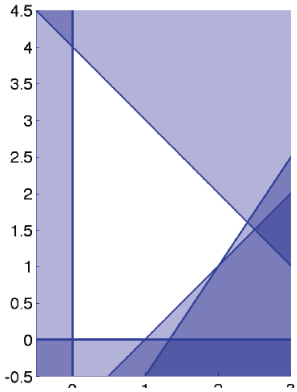
$$\begin{array}{cccccccc} x & y & u & v & w & z & \\ 0 & 3 & -2 & 1 & 0 & 0 & 4 \\ 0 & 1 & -1 & 0 & 1 & 0 & 1 \\ 0 & -1 & 2 & 0 & 0 & 1 & 8 \end{array}$$

The new LP is:

$$\max_{x,y} \quad z = y - 3u + 8 \quad \text{s.t.} \tag{13.2}$$

$$\begin{array}{rcl} y + u & \leq & 4 \\ 3y - 2u & \leq & 4 \\ y - u & \leq & 1 \\ y, u & \geq & 0 \end{array}$$

The following figure shows the feasible region of the transformed LP. Note that this transformation not change the feasbility region.



How are the slacks affected? We can observe that the slack variables changed with the above transformation. Initially, the slack variables were u, v, w but now they are x, u, w. Though the initial and the transformed LPs are equivalent, we can intrepret them differently based on the slack variables.

How is the objective affected? The objective needs to be changed accordingly to eliminate the x variables. The objective function is also transformed using the row operation.

We can change the LPs form by changing the slack variables as seen earlier. There are in general many ways to transform an LP using this receipe. For example, there are $\binom{5}{3}$ different ways in our example. How do we choose a tranformation? We will look into this issue in the next lecture. What if there aren't slacks? Nothing changes. We can still use row operations and introduce slack variables.

13.2.7 Bases

In the standard form, bases is a maximal subset of indices such that the corresponding column vectors are independent. The elements of the bases are called basic variables. It is easy to see that there are always m of them. In the example below $\{u, v, w, z\}$ is a bases.

x	y	u	v	w	z	
1	1	1	0	0	0	4
2	5	0	1	0	0	12
1	2	0	0	1	0	5
-2	-3	0	0	0	1	0

Picking a bases and setting the non-basic variables to 0 is equivalent to picking a corner in the inequality form. In this process, each non-basic variable yields a tight inequality. In the next lecture, we will look at a simple yet powerful algorithm (called simplex algorithm) to solve general LPs using row operations and other concepts here.

■