Linear Programming, Lagrange Multipliers, and Duality

Geoff Gordon

Overview

This is a tutorial about some interesting math and geometry connected with constrained optimization. It is not primarily about algorithms—while it mentions one algorithm for linear programming, that algorithm is not new, and the math and geometry apply to other constrained optimization algorithms as well.

The talk is organized around three increasingly sophisticated versions of the Lagrange multiplier theorem:

- the usual version, for optimizing smooth functions within smooth boundaries,
- an equivalent version based on saddle points, which we will generalize to
- the final version, for optimizing continuous functions over convex regions.

Between the second and third versions, we will detour through the geometry of convex cones.

Finally, we will conclude with a practical example: a high-level description of the log-barrier method for solving linear programs.

Lagrange Multipliers

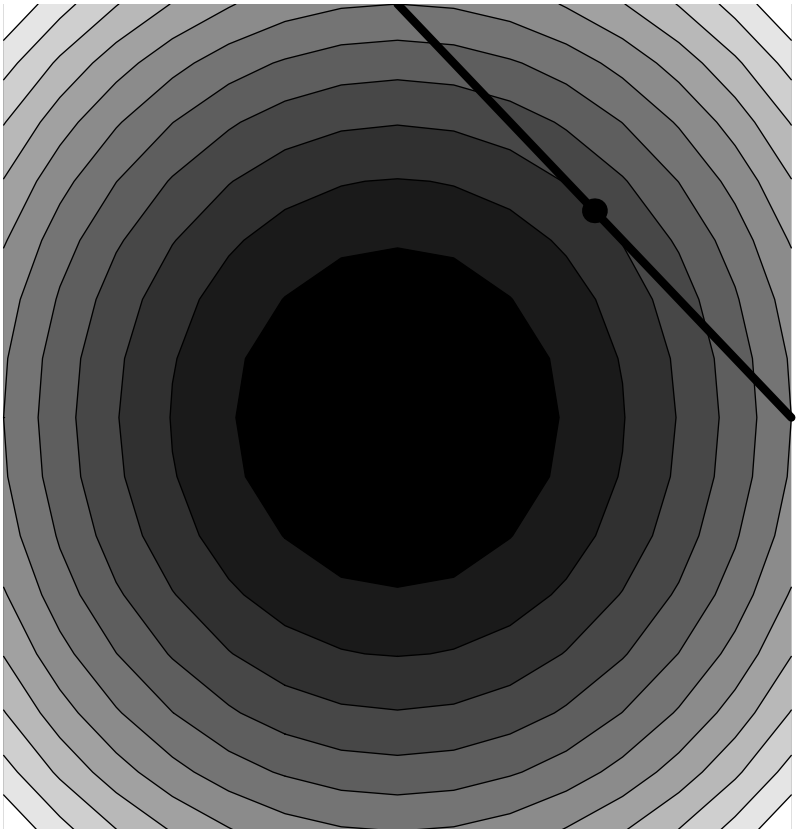
Lagrange multipliers are a way to solve constrained optimization problems. For example, suppose we want to minimize the function

$$f(x, y) = x^2 + y^2$$

subject to the constraint

$$0 = g(x,\ y) = x + y - 2$$

Here are the constraint surface, the contours of f, and the solution.



More Lagrange Multipliers

Notice that, at the solution, the contours of f are tangent to the constraint surface. The simplest version of the **Lagrange Multiplier theorem** says that this will always be the case for equality constraints: at the constrained optimum, if it exists, ∇f will be a multiple of ∇g.

The Lagrange Multiplier theorem lets us translate the original constrained optimization problem into an ordinary system of simultaneous equations at the cost of introducing an extra variable:

$$\begin{aligned} g(x, y) &= 0 \\ \nabla f(x, y) &= p \nabla g(x, y) \end{aligned}$$

The first equation states that x and y must satisfy the original constraint; the second equation adds the qualification that ∇f and ∇g must be parallel. The new variable p is called a **Lagrange multiplier**.

The Lagrangian

We can write this system of equations more compactly by defining the **Lagrangian** L:

$$L(x, y, p) = f(x, y) + p\, g(x, y) \;=\; x^2 + y^2 + p(x + y - 2)$$

The equations are then just $\nabla L = 0$, or in this case

$$\begin{aligned} x + y &= 2 \\ 2\,x &= -p \\ 2\,y &= -p \end{aligned}$$

The unique solution for our example is $x = 1$, $y = 1$, $p = -2$.

Multiple Constraints

The same technique allows us to solve problems with more than one constraint by introducing more than one Lagrange multiplier. For example, if we want to minimize

$$x^2 + y^2 + z^2$$

subject to

$$x + y - 2 = 0$$
$$x + z - 2 = 0$$

we can write the Lagrangian

$$L(x, y, z, p, q) = x^2 + y^2 + z^2 + p(x + y - 2) + q(x + z - 2)$$

and the corresponding equations

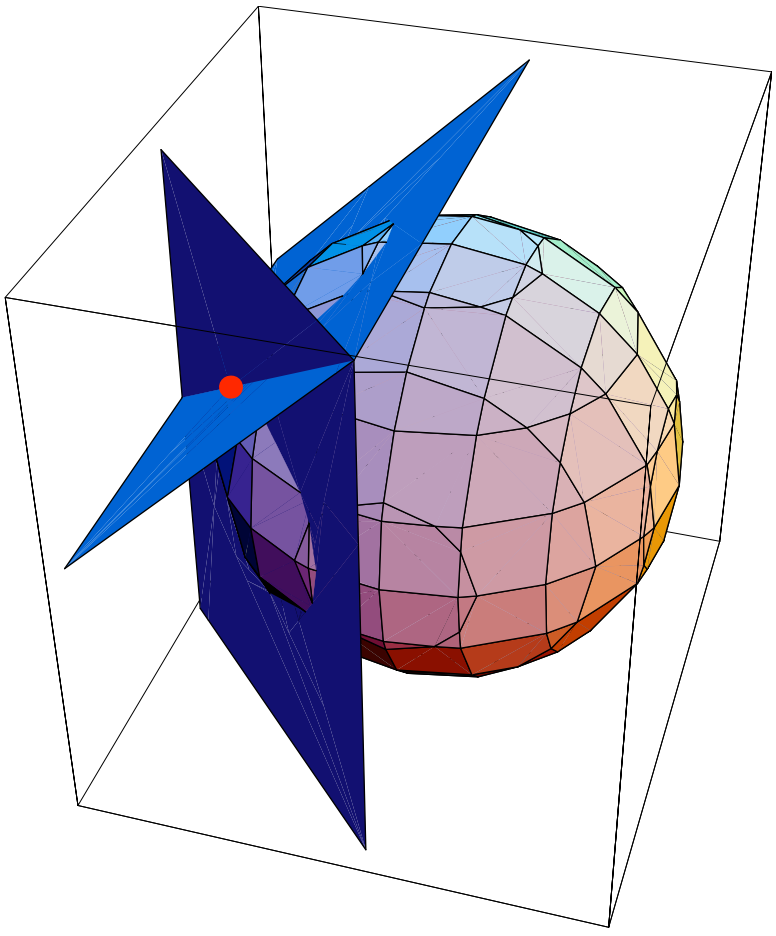
$$0 = \nabla_x L = 2x + p + q$$
$$0 = \nabla_y L = 2y + p$$
$$0 = \nabla_z L = 2z + q$$
$$0 = \nabla_p L = x + y - 2$$
$$0 = \nabla_q L = x + z - 2$$

Multiple Constraints: the Picture

The solution to the above equations is

$$\left\{p \to -\frac{4}{3}, q \to -\frac{4}{3}, x \to \frac{4}{3}, y \to \frac{2}{3}, z \to \frac{2}{3}\right\}$$

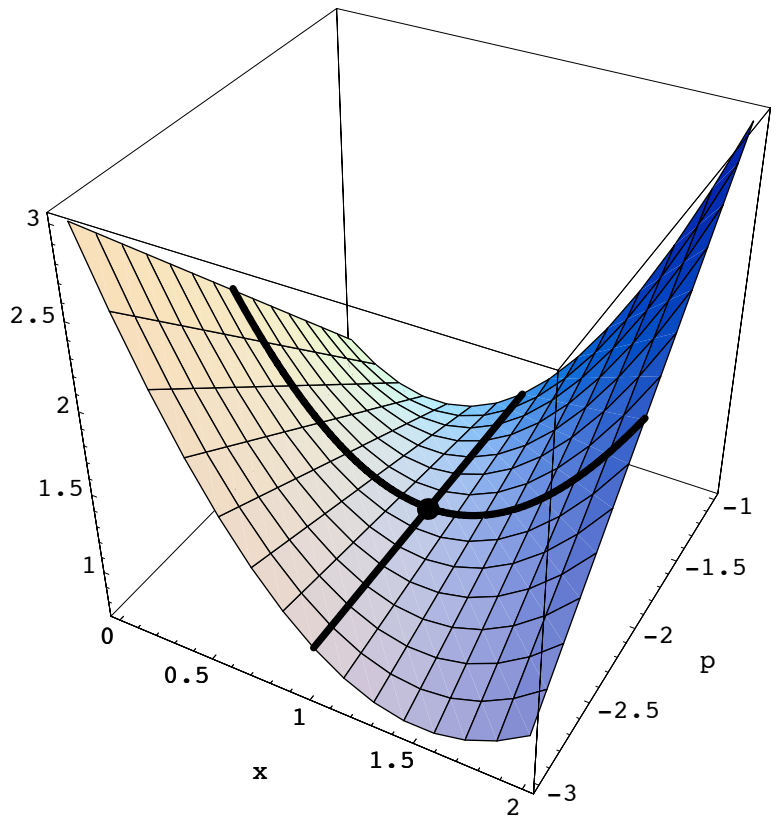
Here are the two constraints, together with a level surface of the objective function. Neither constraint is tangent to the level surface; instead, the normal to the level surface is a linear combination of the normals to the two constraint surfaces (with coefficients p and q).



Saddle Points

Lagrange multiplier theorem, version 2: The solution, if it exists, is always at a saddle point of the Lagrangian: no change in the original variables can decrease the Lagrangian, while no change in the multipliers can increase it.

For example, consider minimizing x^2 subject to $x = 1$. The Lagrangian is $L(x, p) = x^2 + p(x - 1)$, which has a saddle point at $x = 1$, $p = -2$.



For fixed p, L is a parabola with minimum at $-\frac{p}{2}$ (1 when $p = -2$). For fixed x, L is a line with slope $x - 1$ (0 when $x = 1$).

Lagrangians as Games

Because the constrained optimum always occurs at a saddle point of the Lagrangian, we can view a constrained optimization problem as a game between two players: one player controls the original variables and tries to minimize the Lagrangian, while the other controls the multipliers and tries to maximize the Lagrangian.

If the constrained optimization problem is well-posed (that is, has a finite and achievable minimum), the resulting game has a finite value (which is equal to the value of the Lagrangian at its saddle point).

On the other hand, if the constraints are unsatisfiable, the player who controls the Lagrange multipliers can win (i.e., force the value to $+\infty$), while if the objective function has no finite lower bound within the constraint region, the player who controls the original variables can win (i.e., force the value to $-\infty$).

We will not consider the case where the problem has a finite but unachievable infimum (e.g., minimize $\exp(x)$ over the real line).

Inequality Constraints

What if we want to minimize $x^2 + y^2$ subject to $x + y - 2 \geq 0$?

We can use the same Lagrangian as before:

$$L(x, y, p) = x^2 + y^2 + p(x + y - 2)$$

but with the additional restriction that $p \leq 0$.

Now, as long as $x + y - 2 \geq 0$, the player who controls p can't do anything: making p more negative is disadvantageous, since it decreases the Lagrangian, while making p more positive is not allowed.

Unfortunately, when we add inequality constraints, the simple condition $\nabla L = 0$ is neither necessary nor sufficient to guarantee a solution to a constrained optimization problem. (The optimum might occur at a boundary.)

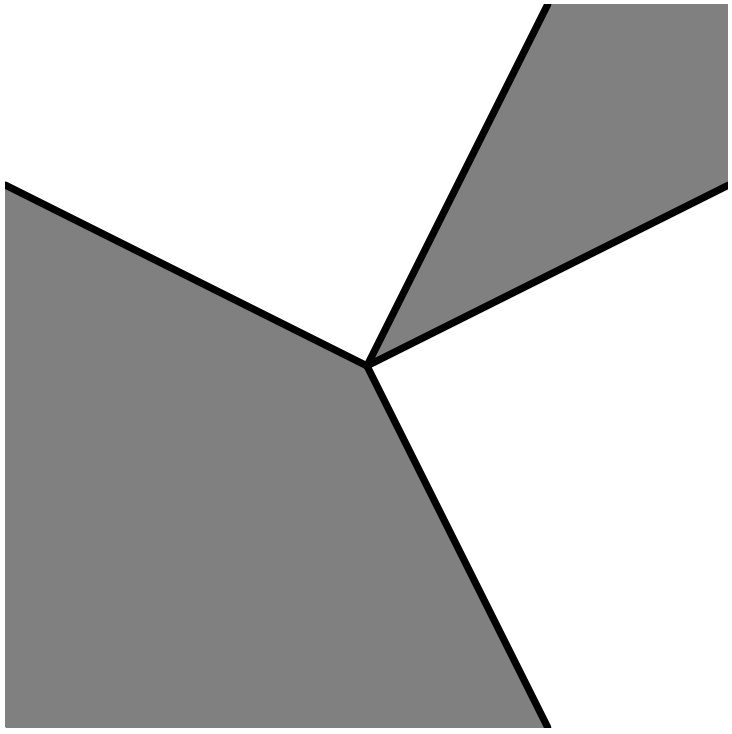
This is why we introduced the saddle-point version of the LM theorem. We can generalize the Lagrange multiplier theorem for inequality constraints, but we must use saddle points to do so.

First, though, we need to look at the geometry of convex cones.

Cones

If $\boldsymbol{g}_1$, $\boldsymbol{g}_2$, ... are vectors, the set $\{\sum a_i \boldsymbol{g}_i \mid a_i \geq 0\}$ of all their nonnegative linear combinations is a **cone**. The $\boldsymbol{g}_i$ are the **generators** of the cone.

The following picture shows two 2-dimensional cones which meet at the origin. The dark lines bordering each cone form a minimal set of generators for it; adding additional generating vectors from inside the cone wouldn't change the result.

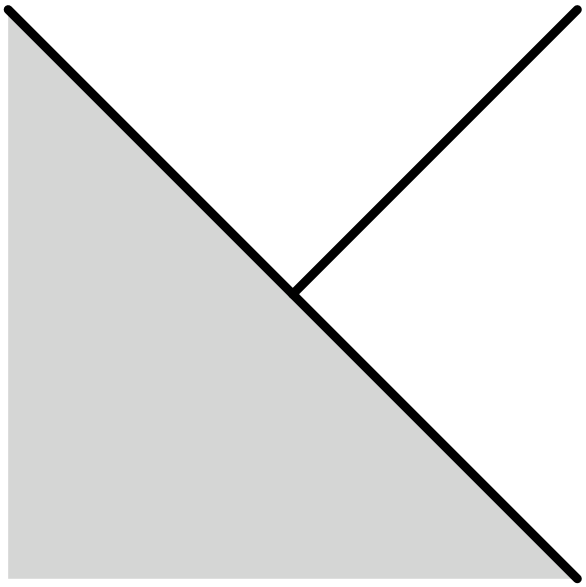


These particular two cones are **duals** of each other. The dual of a set C, written $C^{\perp}$, is the set of all vectors which have a nonpositive dot product with every vector in C. If C is a cone, then $(C^{\perp})^{\perp} = C$.

Special Cones

Any set of vectors can generate a cone. The null set generates a cone which contains only the origin; the dual of this cone is all of $\mathbb{R}^n$. Fewer than n vectors generate a cone which is contained in a subspace of $\mathbb{R}^n$.

If a cone contains both a (nonzero) vector and its negative, it is **flat**. Any linear subspace of $\mathbb{R}^n$ is an example of a flat cone. The following picture shows another flat cone, along with its dual (which is not flat).



A cone which is not flat is **pointed**. The dual of a full-rank flat cone is a pointed cone which is not of full rank; the dual of a full-rank pointed cone is also a full-rank pointed cone.

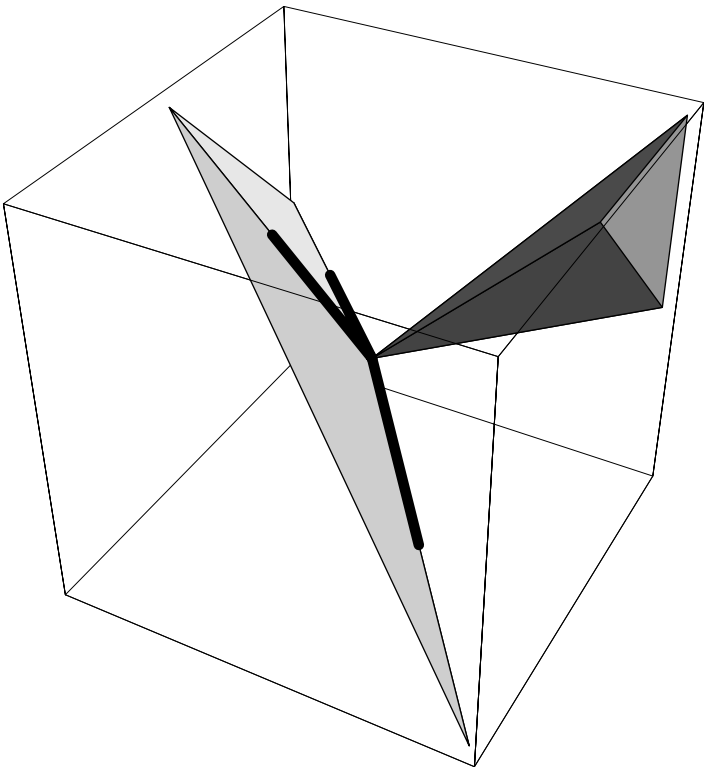
The dual of the positive orthant in $\mathbb{R}^n$ is the negative orthant. If we reflect the negative orthant around the origin, we get back the positive orthant again. Cones with this property (that is, $C = -C^{\perp}$) are called **self-dual**.

Faces and Such

We have already seen the representation of a cone by its generators. Any vector in the minimal set of generators for a cone is called an **edge**. (The minimal set of generators is unique up to scaling.)

Cones can also be represented by their **faces**, or bounding hyperplanes. Since each face must pass through the origin, we can describe a face completely by its normal vector.

Conveniently, the face normals of any cone are the edges of its dual:



Representations of Cones

Any linear transform of a cone is another cone. If we apply the matrix A (not necessarily square) to the cone generated by $\{\boldsymbol{g}_i \mid i = 1..m\}$, the result is the cone generated by $\{A\,\boldsymbol{g}_i \mid i = 1..m\}$.

In particular, if we write G for the matrix with columns $\boldsymbol{g}_i$, the cone C generated by the $\boldsymbol{g}_i$ is just the nonnegative orthant $\mathbb{R}^{m+}$ transformed by G. G is called the **generating matrix** for C. For example, the narrower of the two cones on the previous slide was generated by the matrix

$$\begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \end{pmatrix}$$

On the other hand, we sometimes have the bounding hyperplanes of C rather than its edges. That is, we have a matrix N whose columns are the n face normals of C. In this case, N is the generating matrix of $C^{\perp}$, and we can write C as $(N\,\mathbb{R}^{n+})^{\perp}$. The bounding-hyperplane representation of our example cone is

$$\begin{pmatrix} -1 & -1 & 1 \\ 1 & 0 & -1 \\ 1 & 1 & -1 \end{pmatrix}$$

Unfortunately, in higher dimensions it is difficult to translate between these two representations of a cone. For example, finding the bounding hyperplanes of a cone from its generators is equivalent to finding the convex hull of a set of points.

Convex Polytopes as Cones

A convex polytope is a region formed by the intersection of some number of halfspaces. A cone is also the intersection of halfspaces, with the additional constraint that the halfspace boundaries must pass through the origin. With the addition of an extra variable to represent the constant term, we can represent any convex polytope as a cone: each edge of the cone corresponds to a vertex of the polytope, and each face of the cone corresponds to a face of the polytope.

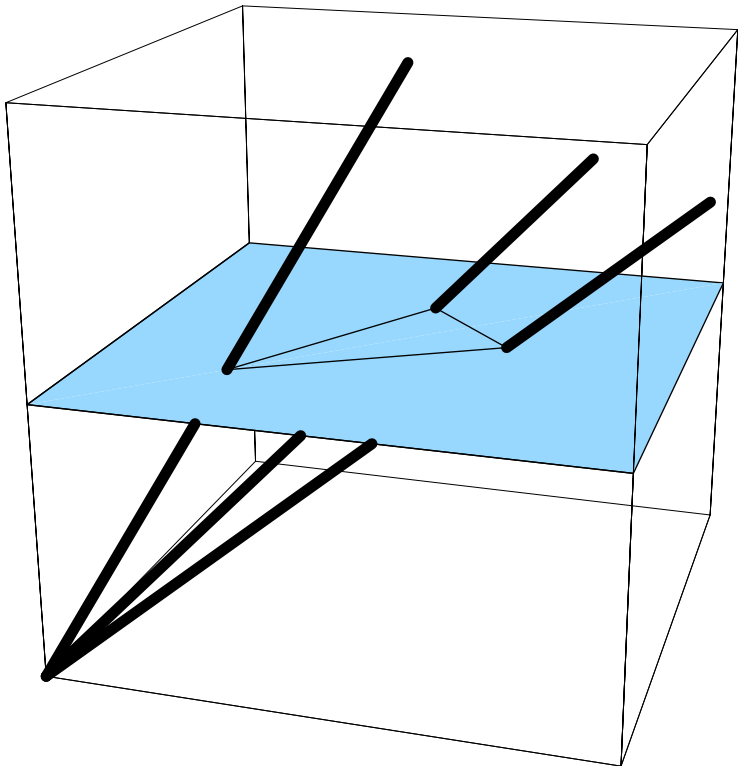
For example, to represent the 2D equilateral triangle with vertices at

$$\begin{pmatrix} 1 \\ 1 \end{pmatrix}\begin{pmatrix} 1+\sqrt{3} \\ 2 \end{pmatrix}\begin{pmatrix} 2 \\ 1+\sqrt{3} \end{pmatrix}$$

we add an extra coordinate (call it t) and write the triangle as the intersection of the plane $t = 1$with the cone generated by

$$\begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}\begin{pmatrix} 1+\sqrt{3} \\ 2 \\ 1 \end{pmatrix}\begin{pmatrix} 2 \\ 1+\sqrt{3} \\ 1 \end{pmatrix}$$

Polytopes cont'd



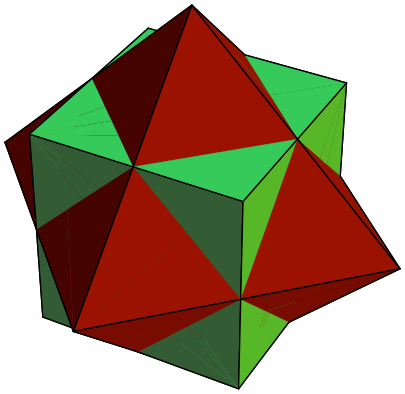
Equivalently, we can represent the same triangle by its bounding hyperplanes, one of which is

$$\begin{pmatrix} \frac{1}{6}\left(-3+\sqrt{3}\right) \\ \frac{1}{6}\left(-3+\sqrt{3}\right) \\ 1 \end{pmatrix} \cdot \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \geq 0$$

We can represent an unbounded polytope by allowing some of the cone's edges not to intersect the plane $t = 1$.

Geometric Duality

The idea of duality for cones is almost the same as the standard idea of geometric duality. A pair of dual cones represents a pair of dual polytopes, with each vertex of one polytope corresponding to a face of the other and vice versa.



Why almost: usually the geometric dual is thought of as insensitive to location and scaling, while the size and location of the polytope represented by the dual cone depend on the size of the original polytope and its position relative to the origin. Also, strictly speaking, each cone must be the negative of the dual of the other, since by convention we intersect with the plane $t = 1$ rather than $t = -1$.

The Lagrange Multiplier Theorem

We can now state the Lagrange multiplier theorem in its most general form, which tells how to minimize a function over an arbitrary convex polytope $N\,\boldsymbol{x} + \boldsymbol{c} \leq 0$.

Lagrange multiplier theorem: The problem of finding $\boldsymbol{x}$ to

minimize $f(\boldsymbol{x})$
subject to $N\,\boldsymbol{x} + \boldsymbol{c} \leq 0$

is equivalent to the problem of finding

$$\arg\min_{\boldsymbol{x}} \max_{\mathrm{p}\in\mathbb{R}^{n+}} \; L(\boldsymbol{x}, \boldsymbol{p})$$

where the Lagrangian L is defined as

$$L(\boldsymbol{x}, \boldsymbol{p}) = f(\boldsymbol{x}) + \boldsymbol{p}^T (N\,\boldsymbol{x} + \boldsymbol{c})$$

for a vector of Lagrange multipliers $\boldsymbol{p}$.

The Lagrange multiplier theorem uses properties of convex cones and duality to transform our original problem (involving an arbitrary polytope) to a problem which mentions only the very simple cone $\mathbb{R}^{n+}$. ($\mathbb{R}^{n+}$ is simple for two reasons: it is axis-parallel, and it is self-dual.)

The trade-offs are that we have introduced extra variables and that we are now looking for a saddle point rather than a minimum.

Linear Programming Duality

If $f(x)$ is a linear function (say $\boldsymbol{f}'\,\boldsymbol{x}$) then the minimization problem is a **linear program**.

The Lagrangian is $\boldsymbol{f}'\,\boldsymbol{x} + \boldsymbol{p}'\,(\boldsymbol{N}\,\boldsymbol{x} + \boldsymbol{c})$ or, rearranging terms, $(\boldsymbol{f}' + \boldsymbol{p}'\,N)\,\boldsymbol{x} + \boldsymbol{p}'\,\boldsymbol{c}$.

The latter is the Lagrangian for a new linear program (called the **dual** program):

maximize $\boldsymbol{p}'\,\boldsymbol{c}$
subject to
$N'\,\boldsymbol{p} + \boldsymbol{f} = 0$
$\boldsymbol{p} \geq 0$

Since $\boldsymbol{x}$ was unrestricted in the original, or **primal**, program, it becomes a vector of Lagrange multipliers for an equality constraint in the dual program.

Note we minimize over $\boldsymbol{x}$ but maximize over $\boldsymbol{p}$. (But changing $\leq$ to $\geq$ changes $\boldsymbol{p}$ to $-\boldsymbol{p}$.)

Duality cont'd

Even though the primal and dual Lagrangians are the same, the primal and dual programs are not (quite) the same problem: in one, we want $\min_{\boldsymbol{p}} \max_{\boldsymbol{x}} L$, in the other we want $\max_{\boldsymbol{x}} \min_{\boldsymbol{p}} L$.

It is a theorem (due to von Neumann) that these two values are the same (and so any $\boldsymbol{x}, \boldsymbol{p}$ which achieves one also achieves the other).

For nonlinear optimization problems, $\inf_{\boldsymbol{p}} \sup_{\boldsymbol{x}} L$ and $\sup_{\boldsymbol{x}} \inf_{\boldsymbol{p}} L$ are not necessarily the same. The difference between the two is called a **duality gap**.

Complementary Slackness

If $\boldsymbol{x}$, $\boldsymbol{p}$ is a saddle point of L, $\boldsymbol{p}^T (N\,\boldsymbol{x} + \boldsymbol{c}) \leq 0$ (else $L(\boldsymbol{x}, 2\,\boldsymbol{p}) > L(\boldsymbol{x}, \boldsymbol{p})$).

Similarly, $\boldsymbol{p}^T (N\,\boldsymbol{x} + \boldsymbol{c}) \geq 0$ (else $L\left(\boldsymbol{x}, \frac{1}{2}\,\boldsymbol{p}\right) > L(\boldsymbol{x}, \boldsymbol{p})$).

So, $\boldsymbol{p}^T (N\,\boldsymbol{x} + \boldsymbol{c}) = 0$ (the **complementary slackness** condition).

Since, in general, $N\,\boldsymbol{x} + \boldsymbol{c}$ will be zero in at most n coordinates, the remaining $m - n$ components of p must be 0. (In fact, barring degeneracy, the constrained optimum will be at a vertex of the feasible region, so $N\,\boldsymbol{x} + \boldsymbol{c}$ will be zero in exactly n coordinates and p will have exactly n nonzeros.)

Prices

Suppose $\boldsymbol{x}$, $\boldsymbol{p}$ are a saddle point of $\boldsymbol{f}' \boldsymbol{x} + \boldsymbol{p}' (N \boldsymbol{x} + \boldsymbol{c})$. How much does an increase of δ in f_i change the value of the optimum?

The value of the optimum increases by at least δx_i, since the same $\boldsymbol{x}$ will still be feasible (although perhaps not optimal).

Similarly, a decrease of δ in c_i decreases the optimum by at least δp_i.

For this reason, dual variables are sometimes called **shadow prices**.

Log-barrier Methods

Commercial linear-programming solvers today use variants of the logarithmic barrier method, a descendent of an algorithm introduced in the 1980s by N. Karmarkar.

Karmarkar's was the second polynomial-time algorithm to be discovered for linear programming, and the first practical one. (Khachiyan's ellipsoid algorithm was the first, but in practice simplex seems to beat it.)

We want to find a saddle point of $L = \boldsymbol{f}' \boldsymbol{x} + \boldsymbol{p}' (N \boldsymbol{x} + \boldsymbol{c})$ with the constraint $\boldsymbol{p} \geq 0$. Since even this simple constraint is inconvenient, consider what happens if we add an extra **barrier term**:

$$L_\mu = \boldsymbol{f}' \boldsymbol{x} + \boldsymbol{p}' (N \boldsymbol{x} + \boldsymbol{c}) - \mu \sum \ln p_i$$

The effect of the barrier term is to make $L_\mu \to \infty$ as $p_i \to 0$, so that we can ignore the constraints on $\boldsymbol{p}$ and work entirely with a smooth function. The relative importance of the barrier term is controlled by the **barrier parameter** $\mu > 0$.

Log-Barrier cont'd

For fixed $\boldsymbol{x}$ and μ, L_μ is strictly convex, and so has a unique minimum. For fixed $\boldsymbol{p}$ and μ, L_μ is linear.

For each μ, L_μ defines a saddle-point problem in $\boldsymbol{x}, \boldsymbol{p}$. For large μ, the problem is very smooth (and easy to solve); as $\mu \to 0$, $L_\mu \to L$ and $(\boldsymbol{x}^*, \boldsymbol{p}^*)_\mu \to \boldsymbol{x}^*, \boldsymbol{p}^*$.

Since L_μ is smooth, we can find the saddle point by setting $\nabla L_\mu = 0$ or

$$N' \boldsymbol{p} + \boldsymbol{f} = 0$$
$$N \boldsymbol{x} + \boldsymbol{c} = \mu \boldsymbol{p}^{-1}$$

where $\boldsymbol{p}^{-1}$ means the componentwise inverse of $\boldsymbol{p}$.

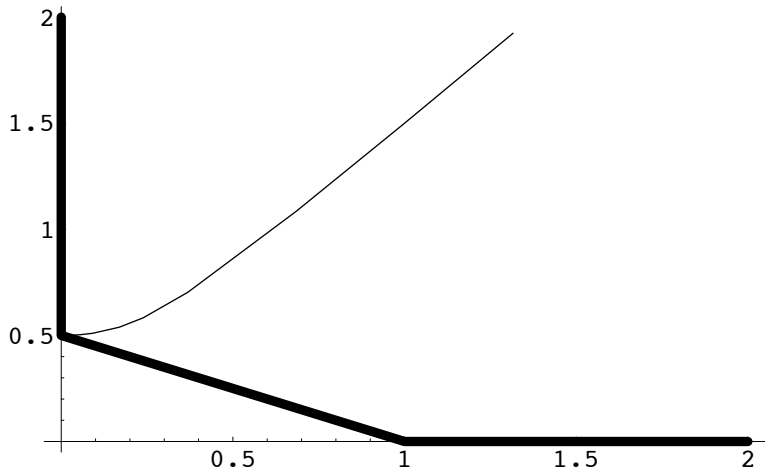
The basic idea of the log-barrier method is to start with a large μ and follow the trajectory of solutions as $\mu \to 0$. This trajectory is called the **central path**.

The Central Path

Here is a portion of the central path for the problem

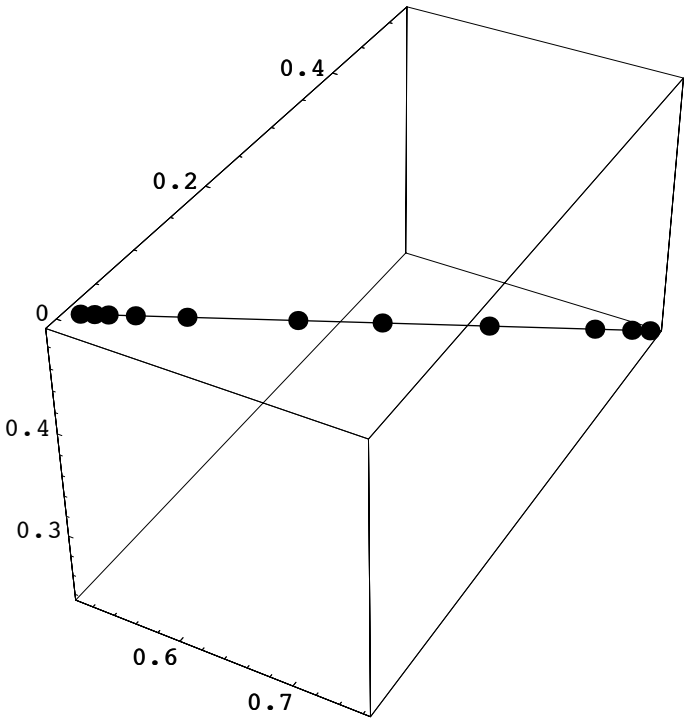
minimize $x + y$
subject to
$x + 2\,y \geq 1$
$x,\ y \geq 0$

This plot shows x, y as a function of μ:



Central Path cont'd

Here are the multipliers p, q, r . Note that p, q, r always satisfy the equality constraints of the dual problem, namely $p + r = 1, q + 2\,r = 1$.



Following the Central Path

Again, the optimality conditions are $\nabla L_\mu = 0$ or

$$N' \boldsymbol{p} + \boldsymbol{f} = 0$$
$$N \boldsymbol{x} + \boldsymbol{c} = \mu \boldsymbol{p}^{-1}$$

If we start with a guess at $\boldsymbol{x}$, $\boldsymbol{p}$ for a particular μ, we can refine the guess with one or more steps of Newton's method.

After we're satisfied with the accuracy, we can decrease μ and start again (using our previous solution as an initial guess). (One could—although we won't—derive bounds on how close we need to be to the solution before we can safely decrease μ.)

With the appropriate (nonobvious!) strategy for reducing μ as much as possible after each Newton step, we have a polynomial time algorithm. (In practice, we never use a provably-polynomial strategy since all the known ones are very conservative.)