Algorithmic Methods 18.10.10

Lecture Notes 1: Introduction to Linear Programming

Professor: Yossi Azar *Scribe: Alexei Kovelman*

1 Linear Programming

Definition: The goal of linear programming (LP) is to find a maximum (minimum) of a given (linear) objective function, given linear constraints (equalities or inequalities). This is deliberately general as many problems can be modeled as linear programming problems; let's take a look at some examples.

2 LP examples

The diet problem: You are the proud owner of a chicken farm and you're given the task of preparing chicken food: there are four brands of chicken food: α, β, γ, δ, each containing a certain percentage of one of the three food groups A,B,C:

	α	β	γ	δ
A	20	9	4	8
B	9	8	18	10
C	10	15	3	9
Price	60	48	40	52

Chicken must consume a certain amount of each food group daily; in our example they must consume (100, 80, 92) daily - that is, 100 units of food group A, 80 of food group B, etc. Each brand also has a price tag (see table above), and given that the farm is on a tight budget, the goal is to minimize the total price of daily feeding while providing the chicken with the necessary food groups.

Now let's describe the problem as an LP problem:

- Our variables are: x_1, x_2, x_3, x_4 - amounts of a brand of food per day for brands α, β, γ, δ.

- Our constraints are:

  1. $20x_1 + 9x_2 + 4x_3 + 8x_4 \geq 100$ (for food group A)
  2. $9x_1 + 8x_2 + 18x_3 + 10x_4 \geq 80$ (B)
  3. $10x_1 + 15x_2 + 3x_3 + 9x_4 \geq 92$ (C)
  4. $x_1, x_2, x_3, x_3 \geq 0$

- Our objective function is: $\min(60x_1 + 48x_2 + 40x_3 + 52x_4)$.

The max-flow problem: Let $G(V, E)$ be a (directed) graph, $s, t \in E$ be the source and sink of the desired flow, let $c : E \to R^+$ be the capacity of the edges of E; we want to find a maximum flow f from source s to sink t that satisfies the following constraints:

- Capacity constraint: $f(u, v) \leq c(u, v) \;\; \forall u, v \in E$.
- Conservation of flows: $\sum_{v:(u,v)\in E} f(u, v) = \sum_{v:(v,u)\in E} f(v, u) \;\; \forall u \in V \backslash \{s, t\}$

The max-flow problem is easy to describe as an LP problem:

- The variables are $f_e \;\; \forall e \in E$ - the flow through each edge, and d - the net flow leaving the source s (and entering the sink t).
- The constraints are:

  1. $f_e \geq 0 \;\; \forall e \in E$ (flow is non-negative).
  2. $f_e \leq c(e) \;\; \forall e \in E$ (capacity constraint).
  3. $\sum_{e:e=(u,v)\in E} f_e - \sum_{e:e=(v,u)\in E} f_e = 0 \;\; \forall u \in V \backslash \{s, t\}$
     (flow conservation for $u \in V \backslash \{s, t\}$).
  4. $\sum_{e:e=(s,v)\in E} f_e - \sum_{e:e=(v,s)\in E} f_e = d$
     (Note: the corresponding equality for the sink t is redundant but may be added).

- The objective function is $\max(d)$.

The multi-commodity max-flow problem: This is a generalization of the max-flow problem:

- We are given a (directed) graph G(V,E) with a capacity function $c : E \to R^+$.
- Rather than have a single source and sink, we have several pairs of sources and sinks $\{s_i, t_i\} \forall i \in \{1...k\}$.
- Each flow from s_i to t_i should be maximized, subject to the rules of flow conservation.

- The *sum* of all flow is subject to constraints of the edges' capacity.

Note: the reduction to a single-source single-sink network using a super-source and a super-sink doesn't work, since it doesn't differentiate between commodities.

This is defined in terms of LP in a similar fashion to the regular max-flow problem:

- The variables are $f_{e,i}$ $\forall e \in E, i \in \{1...k\}$- flow of commodity i through edge e, and d_i for $i \in \{1...k\}$ - net flow of commodity i
- The constraints are:
  1. $f_{e,i} \geq 0$ $\forall e \in E, i \in \{1...k\}$ (flow is non-negative).
  2. $\sum_i f_{e,i} \leq c(e)$ $\forall e \in E$ (capacity constraints for sum of flows).
  3. $\sum_{e:e=(u,v)\in E} f_{e,i} - \sum_{e:e=(v,u)\in E} f_{e,i} = 0$ $\forall u \in V \backslash \{s_i, t_i\}$ and $\forall i \in \{1...k\}$
     (flow conservation for $u \in V \backslash \{s_i, t_i\}$ and commodity i).
  4. $\sum_{e:e=(s_i,v)\in E} f_{e,i} - \sum_{e:e=(v,s_i)\in E} f_{e,i} = d_i$ $\forall i \in \{1...k\}$.
     As before the corresponding inequalities for each sink t_i are redundant but may be added.
- The objective function is $\max(\sum_{i\in\{1...k\}} d_i)$.

This is an easy step from the previous example, but the existing algorithms for the regular max-flow problem (Ford–Fulkerson, Edmonds–Karp, etc.) would have a hard time adapting to the multi-commodity version.

Spam filter: Each mail has a vector of properties (based on its content, its parameters, etc.). We are given a large set of example e-mails already marked as either spam or legitimate correspondence, and based on that information we want to compute a classifier - a hyper-plane that best separates between the spam and the actual mail. Let's define the search for said hyper-plane as an LP problem:

- X_i are samples of valid e-mail, Y_i are spam samples.
- The variables are a_j and b that represent the hyperplane ($\sum x_j a_j = b$).
- The constraints are:
  1. $\forall i \sum_j a_j X_j^i \geq b$ (all valid examples are on the same side of the hyperplane).
  2. $\forall i \sum_j a_j Y_j^i \leq b$ (and all spam is on the other side).
- There is no objective function - we only want to see if such a hyper-plane exists.

3 Different representations of linear programming

3.1 General form:

- Let $A \in R^{kn}$ be a matrix of equalities for k constraints and n variables: $Ax = b$ ($b \in R^k$).
- Let $A' \in R^{ln}$ be a matrix of inequalities for l constraints and n variables: $A'x \geq b'$ ($b' \in R^l$).
- Let r be an integer such that: $x_1 ... x_r \geq 0$ and $x_{r+1} ... x_n \gtreqless 0$ (no constraint on their value).
- The objective function is $\min(c^T x)$ where $c \in R^n$.

3.2 Polyhedral form:

- As above, but only inequalities allowed: $Ax \geq b$.

3.3 Canonical form:

- As in polyhedral form, with the addition of a constraint that $x \geq 0$.

3.4 Standard form:

- A is matrix of equalities: $Ax = b$ such that $x \geq 0$.
- The objective function is $\min(c^T x)$.

3.5 Equivalency of all forms of LP:

We can convert equalities to inequalities in the following manner: given an equality $\overrightarrow{a} \cdot \overrightarrow{x} = b$, convert it into:

- $\overrightarrow{a} \cdot \overrightarrow{x} \geq b$
- $\overrightarrow{a} \cdot \overrightarrow{x} \leq b$

Thus we see that the polyhedral form is equivalent to the general form, and both the canonical and standard forms are special cases of the polyhedral form. In order to prove all-around equivalence, we need two things:

- Represent a polyhedral problem as a canonical problem: we'll represent a variable which could be negative as a difference of two non-negative variables in the following fashion:
  - For all i $x_i = x_i^+ - x_i^-$ where $x_i^+, x_i^- \geq 0$.
- Represent a canonical problem as a standard problem: we convert inequalities into equalities using a dummy variable - given $a_i x \geq b_i$ we add a variable s_i such that $s_i \geq 0$ and convert the inequality into $a_i x - s_i = b_i$.

4 Terms in Linear Algebra

Vector space: a set closed to addition of vectors and multiplication of vector and scalar that satisfies certain axioms (associativity/commutativity of addition, etc).

Linear independence: a set of vectors is independent iff none the vectors in the set can be written as a linear combination of the other vectors in the set.

Dimension: for vector space V, the dimension of V is the cardinality (size) of the largest independent set.

Affine space: given a vector space V, its subspace S and vector $t \in V$, an affine space is defined by $\{y + t | y \in S\}$.

Matrix rank: the dimension of the matrix' rows (columns).

Vector orthogonality: vectors u,v are said to be orthogonal iff $u \cdot v = 0$.

Nullspace: for matrix A, the nullspace (kernel) of A is the set of vectors x such that $Ax = 0$; the nullspace is a linear subspace.

Length: length of vector u is defined as: $||u|| = \sqrt{u \cdot u}$.

Bounded set: a set T is bounded iff there exists v such that $||x|| \leq v \forall x \in T$.

Hyperplane: defined as $\{x | a \cdot x = b\}$.

Half-space: defined as $\{x | a \cdot x \geq b\}$.

Polyhedron: an intersection of a finite number of half-spaces.

Polytope: a bounded, non-empty polyhedron.

Convex combination: a convex combination of vectors X is $\sum_i \lambda_i x_i$ such that $0 \leq \lambda_i \leq 1, \sum_i \lambda_i = 1$.

Strictly convex combination: as above, but: $0 < \lambda_i < 1$.

Convex set: a set S is said to be convex if any convex combination of vectors in S is contained in S (it is sufficient to show for two vectors).

Intersection of convex sets is a convex set.

Polyhedron vertex: a vector in a polyhedron that cannot be written as a strictly convex combination of any *other* vectors in the polyhedron.

5 Theorems

Theorem A: Let $P = \{x \in R^n | Ax \geq b\}$ be a polyhedron; x is a vertex of P if and only if x is feasible and the number of (linearly independent) constraints satisfied tightly in x is exactly n.

Proof: Assume that x is a vertex, and the B is the matrix of tightly satisfied constraints; if rank(B)=n we're done, otherwise rank(B)<n, so there exists a non-zero vector y orthogonal to B ($By = 0$), then if $Bx = b'$ also $B(x + \gamma y) = Bx + \gamma By = b' + 0 = b'$, meaning that for $x + \gamma y$ the constraints are still tightly satisfied. Every constraint not in B, for γ small enough, are still satisfied, so $x + \gamma y$ is feasible, and so is $x - \gamma y$; thus x can be written as a strictly convex combination of $x + \gamma y$ and $x - \gamma y$, contradicting x being a vertex.
Now assume that the number of tightly satisfied constraints in x is n (and they are linearly independent); let B be the matrix of rightly satisfied constraints such that $Bx = b'$; let's assume that x can be expressed as a strictly convex combination $x = \lambda u + (1 - \lambda)v$ for $u, v \in P/\{x\}$, then: $b' = Bx = B(\lambda u + (1 - \lambda)v) = \lambda Bu + (1 - \lambda)Bv \geq \lambda b' + (1 - \lambda)b' = b'$; the inequality holds since u,v are feasible, but since we ended up with an equality, it must be that: $Bv = b', Bu = b'$ (we can apply the inequality separately for u and v); rank(B) = n and hence B is invertible, therefore $x = u, v$, contradicting our assumption.

Corollary 1: Every polytope has a finite number of vertices (given m constraints and n variables, there could be at most $\binom{m}{n}$ vertices).

Theorem B: Let $P = \{x \in R^n | Ax \geq b\}$ be a polytope (bounded and not empty); given an objective function $\min(c^T x)$, for each $x \in P$ there exists a vertex $x' \in P$ such that $c^T x' \leq c^T x$.

Proof: If x is a vertex - we're done; otherwise, the number of tightly satisfied constraints in x is less than n; let B be the matrix of tightly satisfied constraints, as before, and $Bx = b'$; rank(B)<n, so there exists a non-zero y orthogonal to B i.e., $By = 0$. For $x + \gamma y$ the constraints that were satisfied for x are still tightly satisfied: $B(x + \gamma y) = Bx + \gamma By = b' + 0 = b'$. Now let's assume that $c^T y \leq 0$ (meaning that y improves the result - otherwise take $y = -y$). Hence for $\gamma \geq 0$ we have $c^T(x + \gamma y) = c^T x + \gamma c^T y \leq c^T x$; let γ be the maximum value such that $(x + \gamma y) \in P$ (such γ exists since P is a polytope and is bounded) - for that γ the vector $(x + \gamma y)$ satisfies tightly an additional constraint (in addition to those in B). Since the number of constraints is bounded we continue this process until rank$(B) = n$ and we are done.

Corollary 2: A polytope has at least one vertex.

Corollary 3: A finite algorithm that solves LP problems exists (one can look at all the vertices assuming the solution is bounded).

6 Basic feasible solution

Definition: we are given an LP problem in its standard form; A_{mxn} is the matrix of equalities; assume that the rows of A are linearly independent (since we're dealing with equalities, we remove the rows that are linearly dependent with other rows without affecting the result); also assume that $m < n$ (otherwise the solution is exactly one point); rank$(A) = m$; $f = \{x \geq 0 | Ax = b\}$ is the feasible region of A. We take m linearly independent columns and suppose that the other columns are assigned the value of 0; we then solve the remaining mxm set of linear equations and recover the (unique) solution x; such x is called a *basic solution* of A; if $x \geq 0$ then x is feasible and a vertex and will be called *basic feasible solution* (bfs for short). Each bfs has at least $n - m$ zeroes; if a bfs has more that $n - m$ zeros, it will be called a *degenerate bfs*.

Claim: A bfs that is the result of two different selections of columns in a standard-form LP problem is degenerate.

Proof: Let J_1, J_2 be two different selections of non-zero columns in a standard-form LP problem; let x be the solution to both J_1 and J_2; the number of non-zeros in x is at most the size of the intersection of J_1 and J_2 - and that size is strictly less that m, making x degenerate.

Theorem C: In a standard-form LP problem x is a vertex if and only if x is a bfs.

The proof appears in the notes of the next lecture.