IE 495 Lecture 24

November 28, 2000

Reading for This Lecture

- Primary
  - Bazaraa, Sherali, and Sheti, Chapter 2.
  - Chvatal, Chapters 6 and 7.

Linear Programming

Introduction

- Consider again the system $Ax = b$, $A \in \mathbf{R}^{m\times n}$, $b \in \mathbf{R}^m$.
- In this problem, there are either
  - no solutions
  - one solution
  - infinitely many solutions (if n > m)
- The problem of *linear programming* is

$$
\begin{array}{ll}
\min & c^{\mathrm{T}}x \\
\text{s.t.} & Ax = b \\
 & x \geq 0
\end{array}
$$

Applications of Linear Programming

- Linear programming is central to much of operations research.
- Many resource allocation problems can be described as linear programs.
- Example: The Diet Problem
  - We have a set of nutrients with RDAs.
  - We have a set of available foods.
  - We have preference constraints which limit the intake of particular foods.
  - We want to minimize our cost.

Convex Sets

A set S is *convex*

$$\Leftrightarrow$$

$$x_1, x_2 \in S, \lambda \in [0,1] \Rightarrow \lambda x_1 + (1 - \lambda)x_2 \in S$$

- If $x = \Sigma\lambda_i x_i$, where $\lambda_i \geq 0$ and $\Sigma\lambda_i = 1$, then x is a *convex combination* of the x_i's.
- If the positivity restriction on λ is removed, then y is an *affine combination* of the x_i's.
- If we further remove the restriction that $\Sigma\lambda_i = 1$, then we have a *linear combination*.

Extreme Points and Directions

- If S is a convex set in $\mathbf{R}^n$, $x \in S$ is an *extreme point* if it is not a *non-trivial* convex combination of two distinct members of S.
- A vector $d \in \mathbf{R}^n$ is a feasible direction for S if for each $x \in S$, $x + \lambda d \in S \ \forall \lambda \geq 0$
- Notice that this is equivalent to $Ad = 0$, $d \geq 0$.
- A vector $d \in \mathbf{R}^n$ is an *extreme direction* of S if it is not a *non-trivial* convex combination of two distinct feasible directions.

Polyhedral sets

- A polyhedral set is the intersection of a finite number of closed half-spaces, i.e. $\{x \in \mathbf{R}^n \text{ s.t. } Ax \leq b\}$.
- Notice that polyhedral sets are convex.
- Also notice that $\{x \in \mathbf{R}^n \text{ s.t. } Ax = b, x \geq 0\}$ is polyhedral.
- Every element of a polyhedral set is the convex combination of extreme points plus positive scalar multiples of extreme directions.
- A convex set is *bounded* if the set of feasible directions is empty, i.e. if it is the *convex hull* of its extreme points.

Linear Programming and Convexity

- For the remainder of the lecture, we are given $A \in \mathbf{R}^{m \times n}$, $b \in \mathbf{R}^m$. Assume $S = \{x \in \mathbf{R}^{\mathrm{n}} \text{ s.t. } Ax = b,\ x \geq 0\}$ is bounded.
- We want to solve the LP

$$\begin{array}{ll} \min & c^{\mathrm{T}}x \\ \text{s.t.} & Ax = b \\ & x \geq 0 \end{array}$$

- We need only consider extreme points (why?).

Characterization of Extreme Points

- Arrange the columns of A such that $A = [B, N]$, where B is a non-singular $n \times n$ matrix.
- Then x is an extreme point of S if and only if $x = [x_B, 0]$ where $x_B = B^{-1}b$ for some arrangement such that $B^{-1}b \geq 0$
- This implies that the number of extreme points is finite (but still potentially very large).

The Simplex Algorithm

- Note that $x_B = B^{-1}b - B^{-1}Nx_N$
- Hence, $c^T x = c_B{}^T x_B + c_N{}^T x_N = c_B{}^T B^{-1} b + (c_N{}^T - c_B{}^T B^{-1} N) x_N$
- So if $c_N{}^T - c_B{}^T B^{-1} N \geq 0$, we have found the optimal solution (why?).
- Otherwise, suppose some component of $c_N{}^T - c_B{}^T B^{-1} N$ is negative.
- Then we raise the value of the corresponding variable as much as possible while maintaining feasibility.

More Terminology

- The matrix B is called the *basis*.
- The variables corresponding to the columns of B are the *basic* variables.
- All other variables are called *non-basic*.
- The fundamental step the simplex algorithm is called a *pivot*.
  - We add one basic variable and remove another.
  - We do this in such a way that feasibility is maintained and the cost deceases at each step.

Summary of the Simplex Algorithm

- Simplex algorithm
  - Compute $yB = c_B^{\mathrm{T}}$
  - Choose a column of a_j of N such $ya_j > c_j$
  - Compute $Bd = a_j$
  - Find the largest t such that $x_B^* - td \geq 0$
  - Set the value of x_j to t and the values of the basic variables to $x_B^* - td$.
  - Update the basis.
- The only hard part is implementing the last step.

Implementing the Algorithm

- Let B_k be the basis after the k^{th} iteration.
- Note that $B_k = B_{k-1}E_k$ where
  - E_k is the identity matrix with the p^{th} column replaced by $d = B_{k-1}^{-1} a_j$ (already computed).
  - p is the "leaving column"
- So, we have $B_k = B_0 E_1 \dots E_k = LUE_1 \dots E_k$
- To update at each iteration, we merely append the next eta matrix to the list.
- Often, B_0 is the identity matrix.

Refactorizing the Basis

- After many iterations, it can become inefficient to solve these systems.
- Periodically, throw away all the eta files and calculate a brand new LU factorization.
- How often should this be done?
- It depends on the matrix.
- Under some fairly reasonable assumptions, the "break-even" point seems to be $\approx$ 15 iterations.