College of Computer & Information Science
Northeastern University
CS G399: Approxmation Algorithms

Spring 2004
Handout 6
4 March 2004

Introduction To Linear Programming

The following notes are adapted from lecture notes of an Advanced Algorithms course I taught in Fall 1999; these notes were scribed by 3 students Mohsen Ghassemi, Clifford Bryant, Jr., and Trevor Mendez. The material covered is based on Chapter 29 of the text and two other sources: Michel Goemans's lecture notes on linear programming [Goe94], and Howard Karloff's text on linear programming [Kar91].

1 Introduction

We define the linear programming (LP) problem of minimizing a linear function subject to linear inequality constraints. We begin with a simple example of linear programming. Then, the general, standard, and canonical forms of the linear programming problem are given in summation and matrix form. A formal definition is given for a vertex of a polytope or polyhedron, and it is proved that an LP always attains its optimum at a vertex. Then it is proved that the set of vectors corresponding to the current basis of an LP are linearly independent if and only if the basic feasible solution is a vertex point. Finally, we cover the simplex algorithm in detail.

2 The Diet Problem

Many linear programming formulations arise from situations where a decision maker wants to minimize the cost of meeting a set of requirements. In the diet problem we would like to develop a diet using n food items such that it satisfies the daily vitamin requirements. Let the food items be numbered 1 through n and let the m vitamin mineral requirements be given by $b_1, \ldots, b_m$.

From food item j, you get a_{ij} units of mineral i per unit of j. If you decide to have x_j units of item j, you get $x_j a_{ij}$ units of mineral i.

$$\sum_j x_j a_{ij} \geq b_i \quad \forall i$$

If the cost of food item j is c_j per unit. Then the total cost incurred $= \sum_j c_j x_j$.

We would like to minimize $\sum_j c_j \; x_j$
subject to:

$$\sum_{j=1}^{n} x_i a_{i,j} \;\geq\; b_i \quad i = 1, .., m$$

Alternatively, one can write the above optimization problem as:

$$minimize \begin{bmatrix} c_1 & c_2 \dots c_n \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

$$subject\ to: \begin{bmatrix} a_{11} & a_{12} \dots a_{1n} \\ a_{21} & a_{22} \dots a_{2n} \\ a_{m1} & a_{m2} \dots a_{mn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \geq \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}$$

The above problem is of the form min $c^T x$ subject to: $Ax \geq b$. Such that a problem is referenced to as a *linear program*. This is because the objective function as well as the constraints are linear combination of the variables.

We now consider some optimization problems that we have studied and see weather we can write them down as a linear program.

2.1 Knapsack Problem

In the knapsack problem we are given n items of sizes w_1 through w_n and profits p_1 through p_n. The goal is to build the items with total size $\leq B$ such that the total profit is maximized.

The LP representation of the problem can be shown as:

$$\begin{array}{ll} maximize & \sum x_j p_j \\ subject\ to & \sum_{j=1}^{n} x_j w_j \leq B \\ & x_j \in \{0,1\} \end{array}$$

$$c^T = \begin{bmatrix} p_1 \\ \vdots \\ p_n \end{bmatrix}$$

$$A = \begin{bmatrix} w_1 & w_2 \dots w_n \end{bmatrix}$$

$$b = \begin{bmatrix} B \end{bmatrix}$$

Note that the constraint $x_j \in \{0,1\}$ is not a linear constraint so that the above program is an integer linear program.

2.2 Minimum Spanning Tree Problem

In the minimum spanning tree roblem, the set of instances is the set of all weighted undirected graphs. For a given instance graph G, the set $S(G)$ of solutions for G is the set of all trees that span every vertex of G. Finally, the value of a solution tree T is simply the sum of the weights of the edges of T.

One LP representation of the problem is:

$$\begin{array}{lll} \textit{minimize} & \sum_e w_e x_e & \\ \textit{subject to} & \sum_{e \in c} x_e \geq 1 & \textit{for all cut } c \\ & x_e = 0 \textit{ or } 1 & \textit{for all edges } e \end{array}$$

Note that since there are an exponential number of cuts, the number of constraints is exponential.

3 Equivalent Forms

3.1 General Form

This is the general form of the linear programming problem. The c_i's can be interpreted as costs. In this case, the objective is to minimize the total cost subject to the linear constraints.

$$\text{Minimize } \sum_{i=1}^{n} c_i x_i \qquad \text{(objective function)} \tag{1}$$

$$\begin{array}{lrcll} \text{subject to} & \sum_{j=1}^{n} a_{ij} x_j & \geq & b_i & , i = 1, \ldots, m_1 \quad \text{(inequality)} \\ & \sum_{j=1}^{n} a_{ij} x_j & = & b_i & , i = m_1 + 1, \ldots, m_1 + m_2 \quad \text{(equality)} \\ & x_j & \geq & 0 & , j = 1, \ldots, n_1 \quad \text{(non - negativity)} \\ & x_j & \gtreqless & 0 & , j = n_1 + 1, \ldots, n \quad \text{(unconstrained)} \end{array}$$

The general form of the LP can be written more compactly in matrix notation.

$$\text{Minimize } \mathbf{c}^T \mathbf{x} \tag{2}$$

$$\begin{array}{lrcl} \text{subject to} & \mathbf{A}\mathbf{x}_1 & \geq & \mathbf{b} \\ & \mathbf{A}'\mathbf{x}_2 & = & \mathbf{b}' \\ & \mathbf{x}_1 & \geq & 0 \\ & \mathbf{x}_2 & <=> & 0 \end{array}$$

where

$\mathbf{c}$ and $\mathbf{x}$ are n x 1 vectors,

$\mathbf{A}$ is an m_1 x n matrix,

$\mathbf{A}'$ is an m_2 x n matrix,

$\mathbf{b}$ is an m_1 x 1 vector,

$\mathbf{b}'$ is an m_2 x 1 vector,

$\mathbf{x}_1$ is an n_1 x 1 vector,

$\mathbf{x}_2$ is an n_2 x 1 vector, and

$\mathbf{x} = \left\{ \begin{array}{l} \mathbf{x}_1 \\ \mathbf{x}_2 \end{array} \right\}$.

3.2 Standard Form

This is the standard form of the linear programming problem. Here the constraints take the form of linear inequality constraints, plus non-negativity constraints on the independent variables.

$$\begin{aligned} &\text{Minimize } \sum_{i=1}^{n} c_i x_i \qquad \text{(objective function)} \\ \text{subject to } &\textstyle\sum_{j=1}^{n} a_{ij} x_j \geq b_i \quad , i = 1, \ldots, m \quad \text{(inequality)} \\ &x_j \geq 0 \quad , j = 1, \ldots, n \quad \text{(non} - \text{negativity)} \end{aligned} \tag{3}$$

Once more, the canonical form of the LP can be written more compactly in matrix notation.

$$\begin{aligned} &\text{Minimize } \mathbf{c}^T \mathbf{x} \\ \text{subject to } &\mathbf{Ax} \geq \mathbf{b} \\ &\mathbf{x} \geq 0 \end{aligned} \tag{4}$$

where

$\mathbf{c}$ and $\mathbf{x}$ are n x 1 vectors,

$\mathbf{A}$ is an m x n matrix, and

$\mathbf{b}$ is an m x 1 vector.

3.3 Slack Form

This is the slack form of the linear programming problem. Here the constraints take the form of linear equality constraints, plus non-negativity constraints on the independent variables.

$$\begin{aligned} &\text{Minimize } \sum_{i=1}^{n} c_i x_i \qquad \text{(objective function)} \\ \text{subject to } &\textstyle\sum_{j=1}^{n} a_{ij} x_j = b_i \quad , i = 1, \ldots, m \quad \text{(equality)} \\ &x_j \geq 0 \quad , j = 1, \ldots, n \quad \text{(non} - \text{negativity)} \end{aligned} \tag{5}$$

The standard form of the LP can also be written more compactly in matrix notation.

$$\begin{aligned} &\text{Minimize } \mathbf{c}^T \mathbf{x} \\ \text{subject to } &\mathbf{Ax} = \mathbf{b} \\ &\mathbf{x} \geq 0 \end{aligned} \tag{6}$$

where

$\mathbf{c}$ and $\mathbf{x}$ are n x 1 vectors,

$\mathbf{A}$ is an m x n matrix, and

$\mathbf{b}$ is an m x 1 vector.

Definition 1. If x satisfies $\mathbf{Ax} = \mathbf{b}$, $\mathbf{x} \geq \mathbf{0}$, then x is **feasible**.

3.4 Constraint Conversion

It is possible to convert between equality and inequality constraints by the following method.

1. To convert a less than or equal inequality constraint,

$$Ax \leq b$$

to an equality constraint, add the vector of slack variables, s.

$$Ax + s = b,\ s \geq 0$$

2. To convert a greater than or equal inequality constraint,

$$Ax \geq b$$

to an equality constraint, add the vector of surplus variables, t.

$$Ax - t = b,\ t \geq 0$$

3. Finally, to convert an equality constraint,

$$Ax = b$$

to an inequality constraint, add two inequality constraints.

$$Ax \leq b,\ -Ax \leq -b$$

4 Example

Example 1. Consider the following linear program:

$$\begin{array}{lrcrcl} \text{Minimize} & & & x_2 & & \\ \text{subject to} & x_1 & & & \geq & 2 \\ & 3x_1 & - & x_2 & \geq & 0 \\ & x_1 & + & x_2 & \geq & 6 \\ & -x_1 & + & 2x_2 & \geq & 0 \\ & x_1 & & & \geq & 0 \\ & & & x_2 & \geq & 0 \end{array}$$

The optimal solution is (4, 2) of cost 2 (See Figure 1). If we were maximizing x_2, instead of minimizing under the same feasible region, the resulting linear program would be unbounded, since x_2 can increase arbitrarily. From this picture, the reader should be convinced that, for any objective function for which the linear program is bounded, there exists an optimal solution which is a "corner" of the feasible region. This notion will be formalized in the next section.

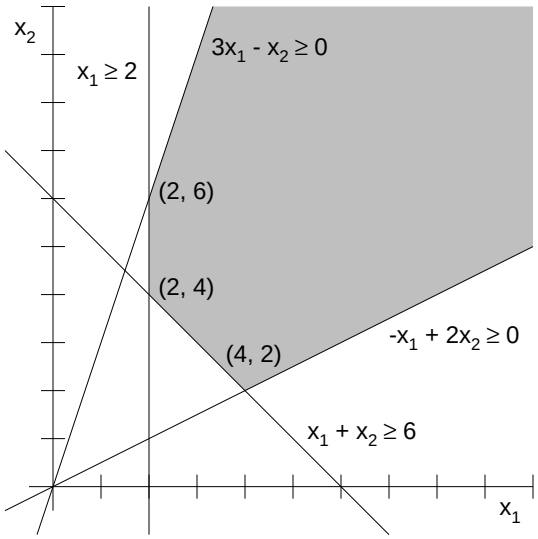

Figure 1: Feasible Region

5 The Geometry of LP

Definition 2. A **polytope** is the analogue in n-dimensional space of point, segment, polygon, and polyhedron in spaces of dimensions 0, 1, 2, and 3. A **convex polytope** in n-space is the convex span of a finite set of points that do not all lie in the same hyperplane; thus a convex polytope is a bounded convex subset enclosed by a finite number of hyperplanes.

Definition 3. A set is **convex** if it contains the line segment joining any two of its points; in a *vector space*, a set such that $rx + (1-r)y$ is in the set for $0 < r < 1$, if x and y are in the set.

Definition 4. A point x is a **vertex** of $\mathcal{P}$ if $\nexists y \neq 0$ such that $x + y,\ x - y \in \mathcal{P}$.

Theorem 1. *If the optimal solution to an LP in canonical form is bounded (from below), then given any $x \in \mathcal{P}$, $\exists$ a vertex v such that $c^T v \leq c^T x$.*

Corollary 1. *If the LP is bounded, then $\exists$ a vertex that yields the optimal solution.*

Proof: (of theorem) Given $x \in \mathcal{P}$. If x is a vertex, we are done. Otherwise, $\exists y \neq 0$ such that $x + y \in \mathcal{P}$ and $x - y \in \mathcal{P}$. By feasibility of $x + y$ and $x - y$, $A(x + y) = b,\ A(x - y) = b$. But this implies that $Ay = 0$. Furthermore, since $x + y \geq 0$ and $x - y \geq 0$, $y_j = 0$ whenever $x_j = 0$.

Without loss of generality, choose y such that $c^T y \leq 0$. Then $c^T(x + \lambda y) = c^T x + \lambda c^T y \leq c^T x$.

Case 1. $\exists j$ such that $y_j < 0$.

We need to ensure that $x + \lambda y \geq 0$ to maintain feasibility. We know that whenever $x_i = 0$, we have $y_i = 0$. Hence set $\lambda = \min_{\{j: y_j < 0\}}\{\frac{x_j}{-y_j}\}$. Then it is clear that $x + \lambda y \geq 0$. Then $x + \lambda y$ has at least one more zero component than x.

Case 2. $y_j \geq 0,\ \forall j$.

In this case, if $c^T y = 0$, then we can negate every component of y and switch to case 1 (note that $y \neq 0$). It remains to consider the subcase $c^T y < 0$. Clearly, $x + \lambda y \geq 0$ for $\lambda > 0$. Hence, $c^T(x + \lambda y) = c^T x + \lambda c^T y$. Then λ can be chosen arbitrarily large, and the objective function is unbounded. Contradiction.

Case 1 can happen at most n times, since x has n components. By induction on the number of non-zero components of x, we obtain a vertex v. ■

Remark 1. Linear Programming was developed during World War II to solve logistics problems for military resources. The simplex method for solving linear programming problems was developed in 1947 by George Dantzig, an American mathematician. Although the simplex method performs well in practice, in 1972 Klee and Minty demonstrated an example in which the simplex method takes an exponential number of steps. In 1979, L. G. Khachian produced the ellipsoidal method, which was shown to solve any linear program in polynomial time. However, when implemented, his method was not competitive with the simplex method, and lost favor. Then, in 1984, Narinder K. Karmarkar presented a method based on the projective transformation of a simplex. Karmarkar showed his method solves any linear program in time that is a polynomial function of the data of the problem. In contrast to Khachian's method, implementations of various generalizations of Karmarkar's method are very competitive with the simplex method, and generally outperform it for large problems.

Theorem 2. *Let x be a point in $\mathcal{P}$. The submatrix $A_x = \{$ columns corresponding to $x_j > 0\}$ is linearly independent if and only if x is a vertex.*

Proof:

($\Leftarrow$) Suppose A_x has linearly dependent columns. Then $\exists y$ such that $A_x y = 0,\ y \neq 0$. Extend y to $\mathbb{R}^n$ by adding zero-valued components. Then $\exists y \in \mathbb{R}^n$ such that $Ay = 0,\ y \neq 0$. Consider $x + \lambda y$ for small $\lambda > 0$. Then we can chose λ such that $x + \lambda y,\ x - \lambda y \in \mathcal{P}$, by an argument analogous to that in Case 1 of the proof of Theorem 1, above. Hence, x is not a vertex.

($\Rightarrow$) Assume x is not a vertex. Then, by definition, $\exists y \neq 0$ such that $x + y,\ x - y \in \mathcal{P}$. Let A_y be the submatrix corresponding to the non-zero components of y. As in the proof of Theorem 1,

$$\left.\begin{array}{l} Ax + Ay = b \\ Ax - Ay = b \end{array}\right\} \Rightarrow Ay = 0.$$

Therefore, A_y has dependent columns, since $y \neq 0$. Moreover,

$$\left.\begin{array}{l} x + y \geq 0 \\ x - y \geq 0 \end{array}\right\} \Rightarrow y_j = 0 \text{ whenever } x_j = 0.$$

Therefore, A_y is a submatrix of A_x. Since A_y is a submatrix of A_x, A_x has linearly dependent columns.

■

6 Simplex Algorithm

The *Simplex Algorithm* is based on the fact that the optimal solution to a feasible LP (linear program) can be found at one of the vertices of the polytope defined by the set of constraints. The algorithm starts from an arbitrary vertex represented by a *basic feasible solution* (bfs), and at each iteration uses a technique called pivoting to search for an adjacent vertex with an improved cost to the solution to move to. If no adjacent vertex has an improved cost, then it can be proved that the current vertex represents the optimal solution. The algorithm must search through a set of potentially exponentially many vertices, and as a result is not polynomial in the worst-case. Even so, it performs very well in practice, and was the algorithm of choice for several decades.

(Throughout this section, all LP's are given by: minimize $c^T x$ subject to $Ax = b$ and $x \geq 0$, unless explicitly defined otherwise.)

6.1 Pivoting

Pivoting is the mechanism used to manipulate the basis corresponding to a vertex v's bfs, to find the basis that corresponds to an adjacent vertex. To perform a pivot step we replace one of the m linearly independent columns of the basis corresponding to v with one of the nonbasic columns from A in such a way that we still have m linearly independent columns. This produces the basis of an adjacent vertex, v'. The Simplex Algorithm uses pivoting to examine the neighbors of v until a neighbor is found that corresponds to a feasible (non-negative) solution which is an improvement over the bfs for v. We illustrate how pivoting is done with an example.

Example 2. *Assume that in LP_1,*

$$A = \begin{bmatrix} 5 & 2 & -3 & 16 & 4 \\ 2 & 3 & 1 & 3 & 1 \\ 1 & 7 & 6 & -1 & 2 \end{bmatrix}$$

$$b = \begin{bmatrix} 8 \\ 8 \\ 25 \end{bmatrix}$$

and we have determined a basis,

$$B = \{1, 3, 5\}.$$

(B is specified here by identifying the basic columns. In other words, $x_2 = x_4 = 0$.) This gives us the following three linearly independent equations, which can be solved to find the corresponding bfs:

$$\begin{aligned} 5x_1 - 3x_2 + 4x_5 &= 8 \\ 2x_1 + x_2 + x_5 &= 8 \\ x_1 + 6x_3 + 2x_5 &= 25 \end{aligned}$$

The solution, $x_1 = 1$, $x_3 = 3$, $x_5 = 3$; can also be written as:

$$bfs = \begin{bmatrix} 1 \\ 0 \\ 3 \\ 0 \\ 3 \end{bmatrix}.$$

We now perform a pivot to find another basis, by adding a nonbasic column to B, and removing one of the basic columns from B. Assume that we choose to add column A_4. We can express A_4 as a linear combination of the columns that are in B as follows:

$$\begin{aligned} A_4 &= \alpha A_1 + \beta A_3 + \gamma A_5 \\ 16 &= 5\alpha - 3\beta + 4\gamma \\ 3 &= 2\alpha + \beta + \gamma \\ -1 &= \alpha + 6\beta + 2\gamma \end{aligned}$$

The solution is: $\alpha = 1$, $\beta = -1$, $\gamma = 2$. Therefore, $A_4 = A_1 - A_3 + 2A_5$. Since we want to find a new basis, B', with a feasible solution, we must find a basic column to remove, such that:

- $A_{B'} x_{B'} = b$,
- $x_{B'} \geq 0$,
- *and* $x_2 = 0$.

where $A_{B'}$ refers to the basic columns of A. We have already solved

$$A_1 x_1 + A_3 x_3 + A_5 x_5 = b$$

and want to modify that solution to include A_4 in the basis so that:

$$\begin{aligned} A_1 x'_1 + A_3 x'_3 + A_4 \Theta + A_5 x'_5 &= b \\ x'_1, x'_3, x'_5 &\geq 0 \end{aligned}$$

Since we know that $A_4 = A_1 - A_3 + 2A_5$, we simply need to find $\Theta \geq 0$ such that,

$$A_1(x'_1 + \Theta) + A_3(x'_3 - \Theta) + A_5(x'_5 + 2\Theta) = b.$$

But we know that:

$$A_1 x_1 + A_3 x_3 + A_5 x_5 = b$$

so we have:

$$\begin{aligned} x_1 &= x'_1 + \Theta = 1, \Rightarrow \Theta \leq 1 \\ x_3 &= x'_3 - \Theta = 3, \Rightarrow \Theta \geq -3 \\ x_5 &= x'_5 + 2\Theta = 3, \Rightarrow \Theta \leq \tfrac{3}{2} \end{aligned}$$

We simply choose the smallest non-negative Θ that satisfies these conditions, namely $\Theta = 1 \Rightarrow x_1 = 0$. Thus our new basis is $B' = \{4, 3, 5\}$, and our bfs has $x'_4 = 1, x'_3 = 4, x'_5 = 1$, and we have completed a pivot. □

Let us review the steps involved in pivoting. The idea is to start from a basis, $B = \{1, 2, \ldots, m\}$ and then find a suitable nonbasic column $j \in \{1, 2, \ldots, m\}$ to replace a basic column $l \notin \{1, 2, \ldots, m\}$ in order to arrive at an adjacent basis, B'. This is achieved using the following steps:

1. Assume that the starting basis $B = \{1, 2, \ldots, m\}$, is known.

   B contains m linearly independent columns of the n columns in A. We have already found the bfs for B, bfs_B which satisfies:

   $$\sum_{i=1}^{m} A_i x_i = b \tag{7}$$

   $$x \geq 0 \tag{8}$$

2. Choose a nonbasic column $j \notin \{1, 2, \ldots, m\}$ to add to B.

   (In the Simplex Algorithm, each $j \notin \{1, 2, \ldots, m\}$ can be tried in turn until one is found that results in a bfs with an *improved* cost.)

3. Find the linear dependence of column j on the basic columns, by finding $\alpha_{i,j}$ such that:

   $$A_j = \sum_{i=1}^{m} \alpha_{i,j} A_i. \tag{9}$$

4. Find our options for $x'_j = \Theta$, so that

   $$\sum_{i=1}^{m} A_i x'_i + A_j x'_j = b \tag{10}$$

   We can find the feasible range of values for Θ as follows:

   $$\begin{aligned}
   \sum_{i=1}^{m} A_i x_i + (A_j x'_j - A_j x'_j) &= b \quad \text{(based on Equation 7)} \\
   \sum_{i=1}^{m} A_i x_i + A_j x'_j - \Theta \textstyle\sum_{i=1}^{m} \alpha_{i,j} A_i & \\
   &= b \quad \text{(based on Equation 9)} \\
   \sum_{i=1}^{m} A_i (x_i - \Theta \alpha_{i,j}) + A_j x'_j &= b
   \end{aligned}$$

   So $x'_i = x_i - \Theta \alpha_{i,j}$, and in order for $x'_i \geq 0$, we must have $\Theta \leq \frac{x_i}{\alpha_{i,j}}$.

5. Choose the basic column to $l \in \{1, 2, \ldots, m\}$ to remove from B.

   We want to choose the smallest value of Θ that will result in a feasible solution, so we take Θ to be the

   $$\underbrace{\min}_{\forall \alpha_{i,j} > 0} \frac{x_i}{\alpha_{i,j}}.$$

   l is the value of i associated with our choice of Θ.

   (N.B. If $\forall \alpha_{i,j} \leq 0$, then the solution is unbounded.)

6.2 Tableau Method

The *Tableau Method* provides an efficient means of performing the pivots required for the Simplex Algorithm. The idea is to manipulate A and b so that the basic columns of A become the identity matrix. When this happens, Step 3 of the pivoting process (above)—finding the $\alpha_{i,j}$ coefficients for A_j—becomes trivial, because (based on Equation 9) $\alpha_{i,j} = a_{i,j}$ (where $a_{i,j}$ is the (i,j) entry of A—the member in the *ith* row of A_j).

Example 3. *Consider LP_2 defined by*

$$\begin{aligned} 6 &= x_1 + x_2 + 4x_6 \\ 14 &= -2x_1 + 2x_2 + x_3 - x_4 + x_6 \\ -11 &= x_1 - 2x_2 + x_4 + 2x_6 \\ 7 &= x_1 - 3x_4 + x_5 - 5x_6 \end{aligned}$$

for which we have determined that a basis, $B = \{2,3,4,5\}$. We can represent LP_2 by an $m \times (n+1)$ matrix, a tableau, as follows:

b	A_1	A_2	A_3	A_4	A_5	A_6
6	1	1	0	0	0	4
14	−2	2	1	−1	0	1
−11	1	−2	0	1	0	2
7	1	0	0	−3	1	−5

By performing row operations on the tableau, we can transfrom the columns that represent B into the identity matrix, without affecting the solution space. This transformation yields,

b	A_1	A_2	A_3	A_4	A_5	A_6
6	1	1	0	0	0	4
3	−1	0	1	0	0	3
1	3	0	0	1	0	10
10	10	0	0	0	1	25

There are two possible nonbasic columns that we may consider adding to our basis as part of a pivot. Assume that we choose column 1, so that $j = 1$. Then we have,

$$A_j = A_1 = 1A_2 - 1A_3 + 3A_4 + 10A_5$$

Θ is determined by

$$\Theta = \underbrace{\min}_{\forall \alpha_{i,1} > 0} \frac{x_i}{\alpha_{i,j}} = \frac{1}{3}.$$

(since $\alpha_{i,j} = a_{i,j}$), so that $l = 4$, and the new basis, B', is $\{2,3,1,5\}$.

Row operations are once again performed in order to convert B' to the identity matrix (and update

b to the new bfs b′), yielding

b'	A'_1	A'_2	A'_3	A'_4	A'_5	A'_6
$\frac{17}{3}$	0	1	0	$\frac{-1}{3}$	0	$\frac{2}{3}$
$\frac{10}{3}$	0	0	1	$\frac{1}{3}$	0	$\frac{19}{3}$
$\frac{1}{3}$	1	0	0	$\frac{1}{3}$	0	$\frac{10}{3}$
$\frac{20}{3}$	0	0	0	$\frac{-10}{3}$	1	$\frac{-25}{3}$

The cost function, $c^T x$, is what actually determines which of the nonbasic columns gets chosen to include in the new basis—we must find one which results in decrease in the cost. For any proposed replacement of some column l by some column j, we must simply check that the cost of the new solution is less that the cost of the previous solution. The cost of new solution is given by:

$$\begin{aligned} \textit{new cost} &= \sum_{i=1}^{m} x'_i c_i + x'_j c_j \\ &= \sum_{i=1}^{m} (x_i - \Theta\alpha_{i,j}) + \Theta c_j \\ &= \underbrace{\sum_{i=1}^{m} x_i c_i}_{\textit{old cost}} + \Theta \underbrace{[c_j - \sum_{i=1}^{m} \alpha_{i,j} c_i]}_{\textit{modified cost, } \bar{c}_j} \end{aligned}$$

So if the modified cost, $\bar{c}_j$, is negative, then the new cost will be reduced. □

Let us review how pivoting is performed using the Tableau Method. A tableau, constructed as follows, is used to perform pivots:

$*$	$\bar{c}_1$	$\bar{c}_2$	$\ldots$	$\bar{c}_n$
b_1	$A_{1,1}$	$A_{1,2}$	$\ldots$	$A_{1,n}$
b_2	$A_{2,1}$	$A_{2,2}$	$\ldots$	$A_{2,n}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
b_m	$A_{m,1}$	$A_{m,2}$	$\ldots$	$A_{m,n}$

in which the basic columns of A form the identity matrix, and the each nonbasic column, j, contains the coefficients, $\alpha_{i,j}$. (An additional row, row 0, is used to store the modified costs.)

A pivot is made with the following steps:

1. Find a column j such that $\bar{c}_j < 0$, by examining row 0.

   (If no such column exists, we have found the optimal solution.)

2. Find column l with $\alpha_{l,j} > 0$, that minimizes Θ

   (If all $\alpha_{i,j} \leq 0$ then either the solution is unbounded–unless we didn't have a basis to begin with.)

3. Replace l by j

We have yet to specify how the starting basis is found for an LP. Suppose we are given LP_A: minimize $c^T x$ subject to $Ax \leq b$, and $x \geq 0$. Then we can add one *surplus* variable per constraint, such that:

$$\sum_j a_{i,j} x_j + s_i = b_i, \quad i \in \{1, 2, \ldots, m\}$$

Then our bfs is: $\{x_j = 0, s_i = b_i\}$. If instead of $Ax \leq b$ we have $Ax = b$, then we can similarly add *artificial* variables, such that:

$$\sum_j a_{i,j} x_j + y_i = b_i i \in \{1, 2, \ldots, m\}$$

to form LP_B (for which we can easily construct a basis) for which we minimize $\sum y_i$, for $y_i \geq 0$. LP_A will be feasible iff the optimal solution for LP_B is 0. To get a bfs for LP_A we solve LP_B using the Simplex Algorithm by starting with the bfs, $\{x_j = 0, y_i = b_i\}$, and if the optimal solution for LP_B is 0, then all $y_i's$ are 0 and the x_i' form a bfs for LP_A.

References

[Goe94] M. Goemans. Introduction to Linear Programming. Lecture notes, available from http://www-math.mit.edu/~goemans/, October 1994.

[Kar91] H. Karloff. *Linear Programming.* Birkhäuser, Boston, MA, 1991.