

Linear Programming

Com S 477/577

Dec 4, 2008

1 Introduction

Many problems can be formulated as maximizing or minimizing an objective in the form of a linear function given a set of linear constraints on the resources. Linear programming has long proved its merit as a significant model of numerous problems in allocation, transportation, scheduling, and economics. It has also found applications in many engineering fields from manufacturing to robotics. The continuing growing applications demonstrate the importance of linear programming as a general framework for problem formulation.

Let us begin with some classic examples that have natural linear programming formulations.

EXAMPLE 1 (The diet problem) Polly wonders how much money she must spend on food in order to get all the energy (2,000 kcal), protein (55 g), and calcium (800 mg) that she needs every day. She chooses six foods that seem to be cheap sources of the nutrients and collects her data in the following table of nutritive value per serving.

| Food | Serving size | Energy (kcal) | Protein (g) | Calcium (mg) | Price per serving (cents) |
|-----------------|--------------|------------------|----------------|-----------------|------------------------------|
| Oat meal | 28 g | 110 | 4 | 2 | 3 |
| Chicken | 100 g | 205 | 32 | 12 | 24 |
| Eggs | 2 large | 160 | 13 | 54 | 13 |
| Whole milk | 237 cc | 160 | 8 | 285 | 9 |
| Cherry pie | 170 g | 420 | 4 | 22 | 20 |
| Pork with beans | 260 g | 260 | 14 | 80 | 19 |

She also decides to impose servings-per-day limits on all foods.

| | Servings at most per day |
|-----------------|--------------------------|
| Oatmeal | 4 |
| Chicken | 3 |
| Eggs | 2 |
| Milk | 8 |
| Cherry pie | 2 |
| Pork with beans | 2 |

To design the most economical menu, she speculates about some as yet unspecified menu, consisting of

x_1 servings of oatmeal, x_2 servings of chicken, and so on. She wants to find numbers x_1, x_2, \dots, x_6 that satisfy her self-imposed servings-per-day limits, meet the requirements for energy, protein, calcium, and yet minimize the cost.

This diet problem can be nicely formulated as follows.

$$\begin{aligned}
 \min \quad & 3x_1 + 24x_2 + 13x_3 + 9x_4 + 20x_5 + 19x_6 \\
 \text{subject to} \quad & 0 \leq x_1 \leq 4 \\
 & 0 \leq x_2 \leq 3 \\
 & 0 \leq x_3 \leq 2 \\
 & 0 \leq x_4 \leq 8 \\
 & 0 \leq x_5 \leq 2 \\
 & 0 \leq x_6 \leq 2 \\
 \text{and} \quad & 110x_1 + 205x_2 + 160x_3 + 160x_4 + 420x_5 + 260x_6 \geq 2000 \\
 & 4x_1 + 32x_2 + 13x_3 + 8x_4 + 4x_5 + 14x_6 \geq 55 \\
 & 2x_1 + 12x_2 + 54x_3 + 285x_4 + 22x_5 + 80x_6 \geq 800
 \end{aligned}$$

EXAMPLE 2. (The transportation problem) Quantities a_1, a_2, \dots, a_m , respectively, of a certain product are to be shipped from each of m locations and received in amounts b_1, b_2, \dots, b_n , respectively, at each of n destinations. We assume, of course, the amount shipped is equal to the amount received, i.e., $\sum_{i=1}^m a_i = \sum_{j=1}^n b_j$.

Associated with the shipping of a unit of product from origin i to destination j is a unit shipping cost c_{ij} . It is desired to determine the amounts x_{ij} to be shipped between each origin-destination pair so as to satisfy the shipping requirements and minimize the total cost of transportation.

We can also formulate this problem as a linear program:

$$\begin{aligned}
 \min \quad & \sum_{i,j} c_{ij}x_{ij} \\
 \text{subject to} \quad & \sum_{j=1}^n x_{ij} = a_i \quad \text{for } i = 1, 2, \dots, m. \\
 & \sum_{i=1}^m x_{ij} = b_j \quad \text{for } j = 1, 2, \dots, n. \\
 & x_{ij} \geq 0 \quad \text{for } i = 1, 2, \dots, m \text{ and } j = 1, 2, \dots, n.
 \end{aligned}$$

2 Standard Form of Linear Programming

Any linear program can be transformed into the following standard form.

$$\begin{aligned}
 \min \quad & c_1x_1 + c_2x_2 + \dots + c_nx_n \\
 \text{subject to} \quad & a_{11}x_1 + a_{12}x_2 + \dots + a_{1n}x_n = b_1 \\
 & \vdots \\
 & a_{m1}x_1 + a_{m2}x_2 + \dots + a_{mn}x_n = b_m \\
 \text{and} \quad & x_1 \geq 0, x_2 \geq 0, \dots, x_n \geq 0.
 \end{aligned}$$

or, more compactly,

$$\begin{aligned}
 \min \quad & \mathbf{c}\mathbf{x} \\
 \text{subject to} \quad & \mathbf{A}\mathbf{x} = \mathbf{b} \text{ and } \mathbf{x} \geq \mathbf{0}.
 \end{aligned}$$

In the above a_i 's, b_i 's, and c_i 's are real constants and x_i 's are real numbers such that

$$\mathbf{x} = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}, \quad A = (a_{ij})_{m \times n}, \quad \mathbf{b} = \begin{pmatrix} b_1 \\ \vdots \\ b_m \end{pmatrix}, \quad \mathbf{c} = (c_1, \dots, c_n).$$

Without loss of generality, we always assume that $b_i \geq 0$ for all i . We also assume that $\text{rank}(A) = m$ from now on; otherwise $\text{rank}(A) < m$, either some equations in $A\mathbf{x} = \mathbf{b}$ are redundant and can be eliminated or the system has no solution.

Below we use three examples to describe how to transform other forms of linear programming into the standard form.

EXAMPLE 3. (slack variables) Consider the problem

$$\begin{aligned} \min \quad & c_1x_1 + c_2x_2 + \dots + c_nx_n \\ \text{subject to} \quad & a_{11}x_1 + a_{12}x_2 + \dots + a_{1n}x_n \leq b_1 \\ & \vdots \\ & a_{m1}x_1 + a_{m2}x_2 + \dots + a_{mn}x_n \leq b_m \\ \text{and} \quad & x_1 \geq 0, x_2 \geq 0, \dots, x_n \geq 0. \end{aligned}$$

Introducing new positive variables y_1, \dots, y_m , called *slack variables*, we can convert the inequalities to equations:

$$\begin{aligned} \min \quad & c_1x_1 + c_2x_2 + \dots + c_nx_n \\ \text{subject to} \quad & a_{11}x_1 + a_{12}x_2 + \dots + a_{1n}x_n + y_1 = b_1 \\ & \vdots \\ & a_{m1}x_1 + a_{m2}x_2 + \dots + a_{mn}x_n + y_m = b_m \\ \text{and} \quad & x_1 \geq 0, x_2 \geq 0, \dots, x_n \geq 0 \\ & y_1 \geq 0, y_2 \geq 0, \dots, y_m \geq 0. \end{aligned}$$

The corresponding matrix becomes (A, I_m) , where I_m is the $m \times m$ identity matrix.

EXAMPLE 4. (surplus variables) If the linear inequalities are in the form of

$$a_{i1}x_1 + a_{i2}x_2 + \dots + a_{in}x_n \geq b_i.$$

We can introduce *surplus variables* $y_i \geq 0$ to change the inequalities to their equivalent forms

$$a_{i1}x_1 + a_{i2}x_2 + \dots + a_{in}x_n - y_i = b_i.$$

EXAMPLE 5. (free variables) When one or more variables can take on any real values, the problem can be transformed to the standard form by applying either of two additional simple techniques described below.

Suppose x_1 is free to take on any real value. The first technique writes

$$x_1 = u_1 - v_1$$

and requires $u_1 \geq 0$ and $v_1 \geq 0$. Then substitutes $u_1 - v_1$ everywhere in the linear program. The problem is then expressed in terms of $n + 1$ variables $u_1, v_1, x_2, \dots, x_n$. Note that a redundancy is introduced since a constant added to both u_1 and v_1 does not change x_1 . Nevertheless, this will not hinder the simplex method introduced later.

The second technique eliminates x_1 together with one of the constraint equations. More specifically, if there is a constant which has nonzero coefficient for x_1 , say,

$$a_{i1}x_1 + a_{i2}x_2 + \cdots + a_{in}x_n = b_i,$$

where $a_{i1} \neq 0$. Then this expression is substituted for x_1 everywhere in the linear program. As a result, there remains $m - 1$ constraints and $n - 1$ variables.

3 Fundamental Theorem of Linear Programming

Consider the system of equations

$$A\mathbf{x} = \mathbf{b},$$

where A is an $m \times n$ matrix, \mathbf{x} is $n \times 1$, and \mathbf{b} is $m \times 1$. As before, we can assume $\text{rank}(A) = m$. For notational simplicity, we also assume that the first m columns of A are linearly independent. Denote by B the $m \times m$ matrix formed by these columns. Then B is nonsingular and the equation

$$B\mathbf{x}_B = \mathbf{b}$$

has a unique solution \mathbf{x}_B , an m -vector. Thus $\mathbf{x} = (\mathbf{x}_B, \mathbf{0})^T$ is a solution to $A\mathbf{x} = \mathbf{b}$. This solution is called a *basic solution* with respect to B . The components of \mathbf{x} associated with columns of B are called *basic variables*, which constitute a *basis*.

Note that B can be made up of any m independent columns, say, columns k_1, \dots, k_m of A . The basic variables are accordingly x_{k_1}, \dots, x_{k_m} .

A vector \mathbf{x} satisfying

$$\begin{aligned} A\mathbf{x} &= \mathbf{b} \\ \mathbf{x} &\geq \mathbf{0} \end{aligned}$$

is said to be *feasible* for these constraints. A feasible solution that is also basic is said to be a *basic feasible solution*.

Theorem 1 *Given a linear program in the standard form*

$$\begin{aligned} \min \quad & \mathbf{c}\mathbf{x} \\ \text{subject to} \quad & A\mathbf{x} = \mathbf{b} \\ & \mathbf{x} \geq \mathbf{0} \end{aligned}$$

where A is an $m \times n$ matrix of rank m , the following statements hold:

1. if there is a feasible solution, there is a basic feasible solution.
2. if there is an optimal feasible solution, there is an optimal basic feasible solution.

For a proof of the above theorem, we refer to [1, pp. 19–20]. The theorem reduces the task of solving an LP problem to that of searching over basic feasible solutions. Since there are at most $\binom{n}{m}$ basic solutions, the fundamental theorem yields an obvious but terribly inefficient finite search technique. But there is a much more efficient procedure called the simplex method.

4 The Simplex Method

The idea of the simplex method, developed by G. B. Dantzig in 1947, is to proceed from one basic feasible solution of the constraint set of a problem in standard form to another, in such a way as to continually decrease the value of the objective function until a minimum is reached. It is efficient in moving basic solutions to construct the minimum.

To obtain a firm grasp of how simplex procedure works may be a little too involved at this moment. So let us work through a simple example first and get a basic picture of what is going on inside the procedure. Then we will see how to deal with issues such as pivoting, moving among basic feasible solutions, determining a minimum feasible solution for a general problem.

Example 6.

$$\begin{aligned} \max \quad & 3x_1 + x_2 + 3x_3 \\ \text{subject to} \quad & 2x_1 + x_2 + x_3 \leq 2 \\ & x_1 + 2x_2 + 3x_3 \leq 5 \\ & 2x_1 + 2x_2 + x_3 \leq 6 \\ & x_1, x_2, x_3 \geq 0 \end{aligned}$$

First, we transform the problem into standard form by introducing three non-negative slack variables x_4, x_5, x_6 and changing the maximization problem to a minimization problem.

$$\begin{aligned} \min \quad & -3x_1 - x_2 - 3x_3 \\ \text{subject to} \quad & 2x_1 + x_2 + x_3 + x_4 = 2 \\ & x_1 + 2x_2 + 3x_3 + x_5 = 5 \\ & 2x_1 + 2x_2 + x_3 + x_6 = 6 \\ & x_1, x_2, \dots, x_6 \geq 0 \end{aligned}$$

It is customary to represent the system by its corresponding array of coefficients, or *tableau*,

| | \mathbf{a}_1 | \mathbf{a}_2 | \mathbf{a}_3 | \mathbf{a}_4 | \mathbf{a}_5 | \mathbf{a}_6 | \mathbf{b} |
|--------------|----------------|----------------|----------------|----------------|----------------|----------------|--------------|
| | 2 | 1 | 1 | 1 | 0 | 0 | 2 |
| | 1 | 2 | 3 | 0 | 1 | 0 | 5 |
| | 2 | 2 | 1 | 0 | 0 | 1 | 6 |
| \mathbf{c} | -3 | -1 | -3 | 0 | 0 | 0 | 0 |

where $\mathbf{a}_1, \dots, \mathbf{a}_6$ are columns of the coefficient matrix. The cost function has its coefficients in the last row of the tableau. The tableau corresponds to the basic feasible solution

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 2 \\ 5 \\ 6 \end{pmatrix}$$

of the transformed problem, with cost 0 and variables x_4, x_5, x_6 in the basis.

Now we can increase either of x_1, x_2, x_3 to decrease the cost since their corresponding entries in the last row of the tableau, which corresponds to the coefficients of the cost function, are all negative: $-3, -1, -3$. But how much can we increase, say, x_2 , without violating the constraint that all x_i must be nonnegative? This is determined by

$$\min \left\{ \frac{b_1}{a_{12}}, \frac{b_2}{a_{22}}, \frac{b_3}{a_{32}} \right\} = \min \left\{ \frac{2}{1}, \frac{5}{2}, \frac{6}{2} \right\} = 2.$$

The corresponding entry a_{12} ($\boxed{1}$ below) in the tableau

$$\begin{array}{ccccccc} \boxed{2} & \boxed{1} & 1 & 1 & 0 & 0 & 2 \\ 1 & 2 & \boxed{3} & 0 & 1 & 0 & 5 \\ 2 & 2 & 1 & 0 & 0 & 1 & 6 \\ \hline -3 & -1 & -3 & 0 & 0 & 0 & 0 \end{array}$$

is called a *pivot*. Similarly, we find two other pivots $\boxed{2}$ and $\boxed{3}$ that determine the maximum possible increase on x_1 and x_3 , respectively.

We can choose any of these three pivots above to bring x_1, x_2 or x_3 into the basis to replace x_4 or x_5 (Note that their corresponding columns have 1s appearing in the same rows with the pivots). Suppose we choose a_{12} as the pivot to bring x_2 into the basis and x_4 out. We do the following two steps to complete the pivoting:

1. Divide the first row by a_{12} (which happens to be 1).
2. Subtract multiples of the (normalized) first row from other rows in the tableau to make all entries in the second column but a_{12} zero.

So we have

$$\begin{array}{ccccccc} 2 & 1 & 1 & 1 & 0 & 0 & 2 \\ -3 & 0 & \boxed{1} & -2 & 1 & 0 & 1 \\ -2 & 0 & -1 & -2 & 0 & 1 & 2 \\ \hline -1 & 0 & -2 & 1 & 0 & 0 & 2 \end{array}$$

Essentially, we substituted

$$x_2 = 2 - 2x_1 - x_3 - x_4$$

into every other constraint equation as well as into the cost function, which now becomes

$$-x_1 - 2x_3 + x_4 - 2.$$

New form of the cost function is read out from the last row in the tableau above, except the last entry is negated. We have thus moved to a new basic solution

$$\begin{pmatrix} 0 \\ 2 \\ 0 \\ 0 \\ 1 \\ 2 \end{pmatrix}$$

and decreased the cost from 0 to -2 .

Next we pivot on $\boxed{1}$ in the last tableau and obtain

$$\begin{array}{ccccccc} \boxed{5} & 1 & 0 & 3 & -1 & 0 & 1 \\ -3 & 0 & 1 & -2 & 1 & 0 & 1 \\ -5 & 0 & 0 & -4 & 1 & 1 & 3 \\ \hline -7 & 0 & 0 & -3 & 2 & 0 & 4 \end{array}$$

The basic solution becomes

$$\begin{pmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 3 \end{pmatrix}$$

and the new cost function is

$$-7x_1 - 3x_4 + 2x_5 - 4$$

with the new cost -4 .

At this point we may pivot in either the first or the fourth column. Let us select $\boxed{5}$ in previous tableau as the pivot, which leads to a new tableau

$$\begin{array}{cccccc} 1 & \frac{1}{5} & 0 & \frac{3}{5} & -\frac{1}{5} & 0 & \frac{1}{5} \\ 0 & \frac{3}{5} & 1 & -\frac{1}{5} & \frac{2}{5} & 0 & \frac{8}{5} \\ 0 & 1 & 0 & -1 & 0 & 1 & 4 \\ \hline 0 & \frac{7}{5} & 0 & \frac{6}{5} & \frac{3}{5} & 0 & \frac{27}{5} \end{array}$$

So the basic solution becomes

$$\begin{pmatrix} \frac{1}{5} \\ 0 \\ \frac{8}{5} \\ 0 \\ 0 \\ 4 \end{pmatrix}.$$

And the new cost function is

$$\frac{7}{5}x_2 + \frac{6}{5}x_4 + \frac{3}{5}x_5 - \frac{27}{5}$$

with cost $-\frac{27}{5}$. In performing this pivoting step, the first row is divided by 5 first before its multiples get subtracted from other rows.

The last row of the tableau now has no negative elements. We can no longer decrease the objective function by increasing the value of x_2, x_4, x_5 . Thus $(\frac{1}{5}, 0, \frac{8}{5}, 0, 0, 4)^T$ is the optimal solution of the transformed problem with the objective value of $-\frac{27}{5}$. The original maximization problem has optimal value $\frac{27}{5}$ achieved at $x_1 = \frac{1}{5}$, $x_2 = 0$, and $x_3 = \frac{8}{5}$.

From Example 6 we can see that by continual pivoting, the simplex algorithm moves among a sequence of equivalent forms (under the equality constraints after the introduction of slack variables) of cost functions:

- (1) $-3x_1 - x_2 - 3x_3$,
- (2) $-x_1 - 2x_3 + x_4 - 2$,
- (3) $-7x_1 - 3x_4 + 2x_5 - 4$,
- (4) $\frac{7}{5}x_2 + \frac{6}{5}x_4 + \frac{3}{5}x_5 - \frac{27}{5}$.

Each move decreases the cost. The algorithm stops when all coefficients in the cost function are nonnegative, in which case the optimal basic feasible solution has been found.

References

- [1] D. G. Luenberger. *Linear and Nonlinear Programming*. Addison-Wesley, 2nd edition, 1984.
- [2] V. Chvátal. *Linear Programming*. W. H. Freeman and Company, 1983.