Linear Programming - II

1 Standard Form

The standard form of a linear programming problem is as follows,

$$\min c^T x, \text{ subject to } Ax = b, x \geq 0,$$

where $A \in \mathbb{R}^{m \times n}$, c and $x \in \mathbb{R}^{n \times 1}$ and $b \in \mathbb{R}^{m \times 1}$. Also, $m \leq n$ and $\text{Rank}(A) = m$. Often, framing a linear programming problem in the standard form is an important task itself.

2 Revised Simplex Algorithm

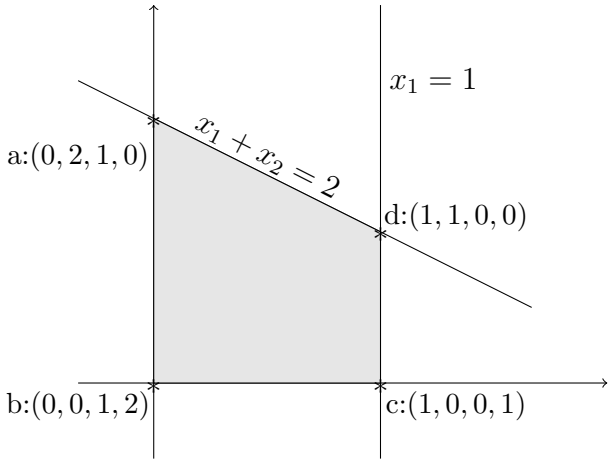

Figure 1: Constraints and vertices of the polytope

Let us consider the following linear programming problem as an example,

$$\min \ x_1 + 2x_2, \text{ subject to } x_1 \leq 1, \ x_1 + x_2 \leq 2, \ x_1 \geq 0, \ x_2 \geq 0$$

Adding slack variables x_3 and $x_4 \in S$ to the constraints to convert to standard form, we get

$$A = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \end{bmatrix}, b = \begin{bmatrix} 1 \\ 2 \end{bmatrix}, c = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$$

Figure: 1 illustrates the constraints and the four vertices of the feasible polytope formed by the constraints of the above example. It can be observed that exactly 2 out of 4 variables are non-zero (active) since $\text{Rank}(A) = 2$ and exactly 1 variable changes to active and 1 variable changes to inactive when we move one vertex to any of its neighbouring vertices.

Also, the solution to this problem lies in one of the vertices of the feasible polytope. This means we have to iterate over n_{C_m} possible vertices to find the optimum solution in a brute force approach. This is combinatorically expensive and so simplex method provides an efficient way of searching this set of vertices.

2.1 Definitions

- $\mathcal{B}$ denotes the subset of index set $\mathcal{I} = \{1, 2, \ldots, n\}$ containing exactly m indices such that $i \notin \mathcal{B} \implies x_i = 0$. This is called the basic set.
- $B = [A_i]_{i \in \mathcal{B}}$ denotes a non-singular $m \times m$ matrix called as the basic matrix.
- $\mathcal{N}$ denotes the non-basic set $\mathcal{N} = \{1, 2, \ldots, n\} \setminus \mathcal{B}$ containing $n - m$ indices such that $i \in \mathcal{N} \implies x_i = 0$.
- $N = [A_i]_{i \in \mathcal{N}}$ denotes a $m \times (n - m)$ matrix called as the non-basic matrix.
- x, c and S are partitioned as basic and non-basic vectors and are defined as below,

$$\begin{array}{ll} x_B = [x_i]_{i \in \mathcal{B}} & x_N = [x_i]_{i \in \mathcal{N}} \\ c_B = [c_i]_{i \in \mathcal{B}} & c_N = [c_i]_{i \in \mathcal{N}} \\ S_B = [S_i]_{i \in \mathcal{B}} & S_N = [S_i]_{i \in \mathcal{N}} \end{array}$$

2.2 Mechanics

The fundamental idea is to move from a feasible vertex to one of its neighbours efficiently and continue to do this till the optimum is reached. In each movement, one index from basic set is swapped with an index from non-basic set. We apply the KKT first order conditions to derive the algorithm. We have

$$Ax = Bx_B + Nx_N = b \text{ and } x_N = 0 \implies x_B = B^{-1}b \tag{1}$$

Also we have from KKT conditions,

$$A^T\lambda + S = c$$

$$B^T\lambda + N^T\lambda + S_B + S_N = c_B + c_N \tag{2}$$

We choose $S_B = 0$ so as to satisfy complementarity KKT condition ($x^T S = 0$). The above equation is then partitioned as,

$$B^T\lambda = c_B \text{ and } N^T\lambda + S_N = c_N$$

$$\implies \lambda = (B^T)^{-1}c_B \tag{3}$$

$$\implies S_N = c_N - (B^{-1}N)^T c_B \tag{4}$$

S_N is called as the *pricing variable* and its components are called as the *reduced costs* of the non-basic variables x_N. They denote the unit decrease in objective function that can be achieved by using the corresponding x_N.

A new index q is chosen from $\mathcal{N}$ such that $s_q \in S_N$ is the most negative value. This ensures that the objective decreases by largest possible value. If $s_q \geq 0, \forall q \in \mathcal{N}$, then the optimum is found and algorithm is terminated. Let x^+ be the new iterate and we have

$$Ax^+ = Bx_B^+ + A_q x_q^+ = Bx_B = Ax$$

$$\implies x_B^+ = x_B - B^{-1}A_q x_q^+ \tag{5}$$

At the new vertex, $x_p = 0, p \in \mathcal{B}$. This index p is removed from $\mathcal{B}$ and moved to non-basic set $\mathcal{N}$. If p is not unique, then we have a degenerate case which is dealt in a different manner not in the scope of this lecture. We will look at how the objective function gets affected by this process. From eqn. 5 we have

$$c^T x^+ = c_B^T x_B^+ + c_q x_q^+ = c_B^T x_B - c_B^T B^{-1} A_q x_q^+ + c_q x_q^+ \tag{6}$$

From eqn. 3, $c_B^T B^{-1} = \lambda^T$ and $A_q^T \lambda = c_q - s_q$. Therefore,

$$c_B^T B^{-1} A_q x_q^+ = (c_q - s_q) x_q^+$$

Substituting the above equation in 6, we get

$$c^T x^+ = c_B^T x_B + c_q x_q^+ - (c_q - s_q) x_q^+$$

$$\implies c^T x^+ = c^T x + s_q x_q^+ \tag{7}$$

Since we chose q such that $x_q \leq 0$, objective is decreased whenever $x_q^+ \geq 0$. If it is possible to increase x_q^+ to ∞ without encountering a new vertex, then the problem is unbounded.

Theorem 1 (13.4 from [2])

If the linear program is nondegenerate and bounded, the simplex method terminates at a basic optimal point.

2.3 Algorithm

The overall algorithm for a single iteration is described below. The inputs are $\mathcal{B}$, $\mathcal{N}$, b, c, x_B and $x_N = 0$ and the outputs are $\mathcal{B}^+$, $\mathcal{N}^+$ and x_B^+.

Algorithm 1 Revised Simplex

function SIMPLEX($\mathcal{B}, \mathcal{N}, b, c$)
  Solve $B^T\lambda = c_B$ for λ.
  Compute $S_N = c_N - N^T\lambda$.
  if $S_N \geq 0$ **then**
    Found Optimal. Terminate algorithm.
  else
    Select $q \in \mathcal{N}$ where s_q is the most negative.
    Solve $Bd = A_q$ for d.
    if $d \leq 0$ **then**
      Unbounded problem. Terminate Algorithm.
    else
      $x_q^+ = \min_{i,d_i \geq 0} \frac{(x_B)_i}{d_i}$. Let p denote the minimizing i. ▷ Ratio Test
      $x_B^+ = x_B - dx_q^+$
      $x_N^+ = (0, \ldots, 0, x_q^+, 0, \ldots, o)^T$
      Change $\mathcal{B}^+ = (\mathcal{B} \setminus \{p\}) \cup \{q\}$, $\mathcal{N}^+ = (\mathcal{N} \setminus \{q\}) \cup \{p\}$
    end if
  end if
end function

It must be noted that the value of objective decreases in most of the steps but not always.

References

[1] George Bernard Dantzig. *Computational algorithm of the revised simplex method*. Rand Corporation, 1954.

[2] Jorge Nocedal and Stephen Wright. *Numerical optimization*. Springer Science & Business Media, 2006.