MIE376
Mathematical Programming

Michael Boyadjian

February 6, 2021

Contents

1 Linear Programming

Linear programming is the field of optimization that concerns (1) the maximization or minimization of a linear function (2) subject to constraints that are also linear.

1.1 Diet Problem

Given a set of foods with associated nutritional information, we want to meet our nutritional requirements while minimizing the cost of the items. We look to find the least cost combination which is a linear problem. We can generalize this to include n food items with m nutritional requirements:

$$x_i = \text{ amount of food item } i$$

$$c_i = \text{ cost of one serving of food item } i$$

$$a_{ij} = \text{ amount of nutrient } i \text{ in one serving of food item } j$$

$$b_j = \text{ amount of nutrient } j \text{ required}$$

Formulating this mathematically as a system of equations, we have the following:

$$\begin{aligned} \textit{minimize}: \quad & c_1x_1 + c_2x_2 + \cdots + c_nx_n \\ \textit{subject to}: \quad & a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n \geq b_1 \\ & a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n \geq b_2 \\ & \qquad\qquad \cdots \\ & a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n \geq b_m \\ & x_1, x_2, \cdots x_n \geq 0 \end{aligned}$$

1.2 Embedded Assumptions

Within linear programming problems, there are four key assumptions we must consider:

- **Proportionality**: The contribution towards the objective function and constraints of decisions are directly proportional to its values
- **Divisibility**: The decision variables can take on any real number
- **Additivity**: The contribution of a decision variable towards the objective or constraints does not depend on other decision variables, the total contribution is the sum of individual contributions of each decision variable
- **Certainty**: The data used as coefficients for a linear programming model such as the objective coefficients c and constraint coefficients A and b are known with certainty.

1.3 General Linear Programming Problems

Linear programming problems can take many forms. They can either maximize or minimize the objective functions. Constraints can be of form $a^T x \leq b$, $a^T x \geq b$, or $a^T x = b$. Variables may also be non-positive, non-negative, or unrestricted. Thus, in general, linear programming problems can be expressed as follows:

$$\begin{aligned} \textit{minimize / maximize}: \quad & c^T x \\ \textit{subject to}: \quad & a_i^T x \geq b_i \quad i \in L \\ & a_i^T x \leq b_i \quad i \in G \\ & a_i^T x = b_i \quad i \in E \\ & x \geq 0 \quad j \in NN \\ & x \leq 0 \quad j \in NP \end{aligned}$$

This could be further expressed in **standard form** as:

$$\begin{aligned} \textit{minimize} \ : \quad & c^T x \\ \textit{subject to}: \quad & Ax = b \\ & x \geq 0 \end{aligned}$$

1.4 Setting up Linear Programming Model

1.4.1 Steps for Model Formulation

In setting up a linear programming model, there are 5 key steps that should be followed:

1. Identify decision variables
2. Form objective function
3. Form constraints
4. Ensure all mathematical expressions are linear
5. Ensure all embedded assumptions are met

1.4.2 Converting to Standard Form

Once the model is formulated, there are then 3 steps in order to convert it to standard form:

1. **Convert Unrestricted Variable**: Set $x = x^+ - x^-$, where $x^+ \geq 0$, $x^- \geq 0$

2. **Convert Inequality Constraints**: For a constraint i, $a_{i1}x_1 + a_{i2}x_2 + \cdots + a_{in}x_n \leq b_i$, add a slack variable, $s_i \leq 0$, so that

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

For a constraint i, $a_{i1}x_1 + a_{i2}x_2 + \cdots + a_{in}x_n \geq b_i$, add a surplus variable, $s_i \geq 0$, so that

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

3. **Convert Maximization to Minimization**: $maximize\ c^T x = -minimize\ -c^T x$

1.5 Terminology

- **Constraint Coefficient**: $A = m \times n$ matrix
- **Cost Coefficient**: $c = n \times 1$ vector
- **Right Hand Side Vector**: $b = m \times 1$ vector
- **Decision Vector**: $x = n \times 1$ vector
- **Feasible Set**: $F = \{x \in \mathbb{R}^n | Ax = b, x \geq 0\}$
- **Optimal Solution**: x^* if vector $x^* \in F$ and $c^T x^* \leq c^T x$ for all $x \in F$
- **Boundedness**: A linear program is **bounded** if $L \leq c^T x$ for all $x \in F$ for some constant L, else it is said to be **unbounded**.

1.6 Absolute Value Problems

If we consider an optimization problem of the following form

$$\begin{aligned} minimize\ : &\quad c_1|x_1| + c_2|x_2| + \cdots + c_n|x_n| \\ subject\ to: &\quad x_i \text{ unrestricted} \qquad i = 1, 2, \cdots n \end{aligned}$$

We can convert this into an equivalent linear program using the following transformations:

$$\begin{aligned} |x_i| &= x_i^+ + x_i^- \\ x_i &= x_i^+ - x_i^- \end{aligned}$$

where x_i^+, $x_i^- \geq 0$. Since the objective is to minimize and $c_i > 0$, then $x_i^+ \times x_i^- = 0$ will hold at optimality.

1.7 Network Optimization Problems

1.7.1 Graph / Network Definitions

Networks can be used to represent many other problems where there is a notion of flow over an entity that can be represented as a network. A network (or graph) G consists of a set of nodes N and a set of edges E (i.e. $G = (N, E)$).

- G is an **undirected** graph when each edge $e \in E$ is an unordered pair of distinct nodes i and j in N. An edge between nodes i and j can be denoted in this case as (i, j) or (j, i).
- G is a **directed** graph when each edge $e \in E$ is an ordered pair of distinct nodes i and j in N. An edge will be denoted in this case as (i, j) and indicates that the direction of the edge is from node i to node j.
- A **path** between nodes i and j of a graph G is a sequence of nodes and edges starting from i and ending with j such that no nodes are repeated. It is important to note that all edges specified in a path must be in E.
- A graph $G = (N, E)$ is **connected** if there is a path between every pair of distinct nodes i and $j \in N$.

1.7.2 Minimum Cost Flow Problem

We want to determine a least cost shipment (flow) of a product (item) through a connected directed network $G = (N, E)$ in order to satisfy demand at various nodes from various supply nodes. There are 3 types of nodes:

- supply node ($b_i > 0$)
- demand node ($b_i < 0$)
- transshipment nodes ($b_i = 0$)

For each $(i, j) \in E$, c_{ij} is the cost of shipping one unit of product from node i to node j. Let x_{ij} be the amount of product to be shipped from node i to node j. The minimum cost flow problem is then:

$$
\begin{aligned}
&\textit{minimize}: \sum_{(i,j)\in E} c_{ij}x_{ij} \\
&\textit{subject to}: \sum_{(i,j)\in E} x_{ij} - \sum_{(j,i)\in E} x_{ji} = b_i \quad \text{for all } i \in N \\
&\qquad\qquad l_{ij} \le x_{ij} \le u_{ij} \quad \text{for all } (i,j) \in N
\end{aligned}
$$

1.7.3 Maximum Flow Problem

The max flow problem is to determine the maximum amount of flow that can be sent from a source node s to a sink node t over a directed network. Let $G = (N, E)$ be a directed graph and u_{ij} be the capacity of edge $(i, j) \in E$. The linear program is thus described as:

$$
\begin{aligned}
\textit{maximize} \ : \ & v \\
\textit{subject to} \ : \ & \sum_{(i,j)\in E} x_{ij} - \sum_{(j,i)\in E} x_{ji} = v \quad \text{for } i = s \\
& \sum_{(i,j)\in E} x_{ij} - \sum_{(j,i)\in E} x_{ji} = 0 \quad \text{for all } i \in N \text{ except} s \text{ and } t \\
& \sum_{(i,j)\in E} x_{ij} - \sum_{(j,i)\in E} x_{ji} = -v \quad \text{for } i = t \\
& l_{ij} \leq x_{ij} \leq u_{ij} \quad \text{for all } (i, j) \in N
\end{aligned}
$$

The maximum flow problem makes the following assumptions:

1. The edge capacities are non-negative and integral
2. There is no directed path from s to t, all of whose edges are uncapacitated (i.e. infinite capacity)
3. There are no parallel edges (i.e. two or more edges with the same starting node i and ending node j)
4. For any edge $(i, j) \in E$, the edge $(j, i) \in E$ as well

2 Linear Programming Geometry

2.1 Geometry of the Feasible Set

We explore additional geometric properties of feasible sets of general linear programs that are consistent, whose feasible set is non-empty. We summarize the key definitions below:

- **Closed Halfspace**: A set of the form $H_{\leq} = \{x \in \mathbb{R}^n | a^T x \leq \beta\}$ or $H_{\geq} = \{x \in \mathbb{R}^n | a^T x \geq \beta\}$; an *inequality constraint*
- **Hyperplane**: A set of the form $H = \{x \in \mathbb{R}^n | a^T x = \beta\}$ where $a \neq 0$ and $\beta \in \mathbb{R}^1$ is scalar; an *equality constraint*. Any hyperplane can be expressed as two closed halfspaces. Therefore, feasible point $x \in P$ lies in the intersection of closed halfspaces.
- **Polyhedron**: The intersection of a finite number of closed halfspaces, also known as the *polyhedral set*
- **Polytope**: A bounded polyhedron
- **Convexity**: A set $C \subseteq \mathbb{R}^n$ is said to be convex if for any x and y in C, $\lambda x + (1 - \lambda)y \in C$ for all $\lambda \in [0, 1]$. This means if we consider two points x and y in $H_{\leq}$ (or $H_{\geq}$), then the line segment from x to y will also be contained in $H_{\leq}$ (or $H_{\geq}$). There are some important theorems that follow from this:
  - **Theorem 2.7**: The closed halfspaces $H_{\leq}$ and $H_{\geq}$ are convex sets
  - **Theorem 2.8**: The intersection of convex sets is convex.
  - **Corollary 2.9**: The feasible set of a linear program is a convex set.

In general, we can characterize optimal solutions to LPs via the geometry of the hyperplane and feasible set intersection.

Let $P \neq \emptyset$ be the feasible set of a linear program with an objective to minimize $c^T x$ and let $H = \{x \in \mathbb{R} | -c^T x = \beta\}$ If $P \subset H_{\leq} = \{x \in \mathbb{R}^2 | -c^T x \leq \beta\}$ for some $\beta \in \mathbb{R}^1$, then any x in the intersection of P and H is an optimal solution for the linear program. There are 3 cases to consider:

- Unique Intersection:
- Infinite Intersection:
- Unbounded Case:

2.2 Extreme Points and Basic Feasible Solutions

We develop an algebraic representation of corner points through the corresponding geometric notion of extreme points. Then, an algebraic representation of extreme points. There are a few key definitions to introduce:

- **Convex Combination:** A convex combinations of vectors x_1, x_2, $\cdots$, $x_k \in R^n$ is a linear combination of $\sum_{i=1}^{k} \lambda_i x_i$ of these vectors such that $\sum_{i=1}^{k} \lambda_i = 1$ and $lambda_i \geq 0$
- **Extreme Point:** Let $C \subseteq \mathbb{R}^n$ be a convex set and $x \in C$. A point x is an extreme point of C if it cannot be expressed as a convex combination of other points in C.

The correspondence between the positive components of an extreme point and the invertibility of the corresponding matrix B turns out to algebraically characterize extreme points. We have the following theorem to express this formally:

Theorem 2.13: Consider a linear program in standard form where the feasible set $P = \{x \in \mathbb{R}^n | Ax = b, x \geq 0\}$ is non-empty. A vector $x \in P$ is an extreme point if and only if the columns of A corresponding to positive components of x are linearly independent.

2.3 Resolution Theorem

2.3.1 Resolutions (Representation) Theorem

We seek a representation of any $x \in P$ is in terms of the extreme points of P and recession directions. We first define a ray and recession direction.

- **Ray:** A ray is a set of the form $\{x \in \mathbb{R}^n | x = x_0 + \lambda d \text{ for } \lambda \geq 0\}$ where x_0 is a given point and d is a non-zero vector called the direction vector.
- **Recession Direction**: Let P be a non-empty feasible set of an LP. A non-zero direction d is called a recession direction if for any $x_0 \in P$, the ray $\{x \in \mathbb{R}^n | x = x_0 + \lambda d \text{ for } \lambda \geq 0\} \subset P$.

2.3.2 Fundamental Theorem of Linear Programming

3 The Simplex Method

4 Duality Theory