# Introduction to LINEAR OPTIMIZATION and EXTENSIONS with MATLAB®

ROY H. KWON

CRC Press
Taylor & Francis Group

# Introduction to LINEAR OPTIMIZATION and EXTENSIONS with MATLAB®

# The Operations Research Series

Series Editor: A. Ravi Ravindran
Professor, Department of Industrial and Manufacturing Engineering
The Pennsylvania State University – University Park, PA

## Published Titles:

Introduction to Linear Optimization and Extensions with MATLAB®
Roy H. Kwon

Supply Chain Engineering: Models and Applications
A. Ravi Ravindran & Donald Paul Warsing

Analysis of Queues: Methods and Applications
Natarajan Gautam

Integer Programming: Theory and Practice
John K. Karlof

Operations Research and Management Science Handbook
A. Ravi Ravindran

Operations Research Applications
A. Ravi Ravindran

Operations Research: A Practical Introduction
Michael W. Carter & Camille C. Price

Operations Research Calculations Handbook, Second Edition
Dennis Blumenfeld

Operations Research Methodologies
A. Ravi Ravindran

Probability Models in Operations Research
C. Richard Cassady & Joel A. Nachlas

# Introduction to LINEAR OPTIMIZATION and EXTENSIONS with MATLAB®

ROY H. KWON

CRC Press
Taylor & Francis Group
Boca Raton London New York

CRC Press is an imprint of the
Taylor & Francis Group, an **informa** business

CRC Press
Taylor & Francis Group
6000 Broken Sound Parkway NW, Suite 300
Boca Raton, FL 33487-2742

No claim to original U.S. Government works
Version Date: 20130725

International Standard Book Number-13: 978-1-4822-0434-6 (eBook - PDF)

**Visit the Taylor & Francis Web site at**
**http://www.taylorandfrancis.com**

**and the CRC Press Web site at**
**http://www.crcpress.com**

*This work is dedicated to my wife Gina and son Noah and my parents Donald and Ester.*

# *Preface*

This book is an outgrowth of lecture notes used for teaching linear programming to graduate students at the University of Toronto (U of T). There have been hundreds of graduate students over the last decade from various parts of the U of T that have taken my courses, most notably from industrial engineering/operations research, electrical, civil, mechanical, and chemical engineering. This group also includes students in the Masters of Mathematical Finance (MMF) Program at the U of T, where I have been teaching a special course in operations research for which the bulk of topics relate to linear and quadratic programming with applications in finance, e.g., portfolio optimization.

Providing concrete examples and illustrations before more general theory seems to work well for most students, and this book aims to take that path. In fact, the book can be used without the need to go through all of the proofs in the book. Students that plan on specializing in optimization would be encouraged to understand all proofs in the book as well as tackle the more theory-oriented exercises. Thus, the material in this book is designed to be of interest and to be accessible to a wide range of people who may be interested in the serious study of linear optimization. This book may be of special interest to those that are interested in financial optimization and logistics and supply chain management. Many of the students regard the computational aspects as an essential learning experience. This has been reflected in this book in that MATLAB® is integrated along with the learning of the conceptual and theoretical aspects of the material.

A unique feature of this book is the inclusion of material concerning linear programming under uncertainty. Both stochastic programming and robust optimization are introduced as frameworks to deal with parameter uncertainty. It is novel to develop these topics in an introductory book on linear optimization, and important, as most applications require decisions to be made in the face of uncertainty and therefore these topics should be introduced as early as possible.

Furthermore, this book is not encyclopedic and is intended to be used in a one-semester course. The main topics were chosen based on a set of core topics that would be needed as well as additional topics that round out and illustrate the modern development of linear optimization and extensions. For example, this book discusses interior point methods but only develops primal-dual path-following methods and not the myriad other interior point methods for linear programming. To this end, we chose the primal-dual path-following method based on its good theoretical and practical properties and yet at the same time illustrates the key issues involved in designing any interior point method.

This book avoids the use of tableaus in the development of the simplex method. Tableaus have been a mainstay for most presentations of the simplex method-based algorithms for linear programming. However, this books takes the view that the underlying geometry of linear programming is such that the algorithms (not just the simplex method) have a natural geometrical analog in the matrix algebra representation version, which is lost in using tableaus. In particular, simplex method-based algorithms are iterative and are viewed naturally as finding a direction of improvement and step length from a current iterate to get to an improved point and so on until optimality is reached or the problem is discovered to be unbounded. A consequence is that it becomes even more natural for MATLAB to facilitate algorithmic understanding by leaving the elementary row operations in performing inversions to MATLAB instead of requiring a student to do the equivalent by doing a pivot on the tableau.

The prerequisites for this book are courses in linear algebra, multi-variate calculus, and basic proficiency in MATLAB. Well-prepared advanced undergraduates could find the book accessible as well. In fact, only several concepts from linear algebra and multi-variate calculus are needed. The appendix contains those concepts from linear algebra that are especially relevant in this book. The multi-variate calculus is reviewed at those points in the book that require it. What I mean by basic MATLAB proficiency is that one knows how to perform standard matrix algebra operations in MATLAB, e.g., multiplying two matrices together and solving a system of linear equations. In any case, MATLAB is a very convenient and powerful platform for optimization and it is relatively easy to get started. A starting point for review are the excellent on-line tutorials and resources available from MathWorks at the website

http://www.mathworks.com/academia/student_center/tutorials
/launchpad.html

Chapter 1 introduces the linear programming problem and gives many examples starting from the well-known diet problem to more complex network optimization models. Various transformation techniques are given so that one can transform an arbitrary linear program in standard form. The MATLAB function linprog is introduced showing how one can solve linear programs on a computer. A computational (case study) project requires the construction and solution of a larger-sized (compared to examples in the chapter) linear program using real financial data in MATLAB.

Chapter 2 develops the geometry of linear programming. First, the geometry of the feasible set of an LP is considered. The geometry of LP gives insight on the nature of optimal and unbounded solutions in terms of corner points, extreme points, and directions of unboundedness. A key development in this chapter is the corresponding algebraic notions of a basic feasible solution and extreme directions. The chapter culminates with the Fundamental Theorem of Linear Programming, which states that the optimal solution for a finite LP can be attained at a basic feasible solution.

Chapter 3 develops the simplex method. As mentioned, the development eschews the tableau construct and the simplex method is developed within the matrix algebraic representation given by the partition of the problem into basic and non-basic variables. Detailed examples are given that illustrate the various possibilities in executing the simplex method including cycling. The revised simplex method is then developed, which brings to light the importance of numerical linear algebra in solving linear programs. MATLAB code is given that implements the simplex method. The MATLAB code is not claimed to be the most efficient or robust, but serves as an example of how the simplex method, as described in the chapter, may be implemented.

Chapter 4 considers duality theory of linear programming. Duality theory enables the development of another variant of the simplex method called the dual simplex method. Economic interpretations of dual variables are discussed and then sensitivity analysis is developed.

Chapter 5 develops the Dantzig-Wolfe decomposition method and illustrates the very important strategy of exploiting structure in a linear programming problem. MATLAB code is given to the illustrate the implementation of the decomposition.

Chapter 6 considers an interior point strategy to solve linear programs. In particular, the class of primal-dual path following methods are developed and then a variant from this class called the predictor-corrector method is considered and implemented in MATLAB.

Chapter 7 develops quadratic programming theory and develops optimality conditions for both unconstrained and constrained versions of the problem. The mean-variance portfolio optimization problem is used as an example of a quadratic program and is featured in several of the numerical examples in the chapter. The MATLAB function quadprog is illustrated, which enables quadratic programs to be solved on computer. An application in generating the efficient frontier of a mean-variance portfolio problem is given. Quadratic programming is discussed in the context of convex optimization. A predictor-corrector interior point method for convex quadratic programming is given.

Chapter 8 considers linear programming under uncertainty. The stochastic programming with recourse framework is developed first. The L-Shaped method is developed to solve two-stage stochastic programs with recourse. Then, robust optimization is developed. Examples of developing robust counterparts are considered and illustrated through a robust portfolio problem. A key theme here is the emphasis on tractable robust formulations.

This book was designed to be used in a semester-long course. Chapters 1 through 4 would be considered as the core part of a course based on this book. The remaining chapters do not have to be considered in a linear order. A course emphasizing interior point methods can cover both Chapters 6 and 7. Parts of Chapter 8 depend on Chapter 5, e.g., the proof of convergence of the L-Shaped method needs the development of the Dantzig-Wolfe decomposition.

This book has nearly 100 exercises and a complete solutions manual is available to instructors. Several of the problems require the use of MATLAB. PowerPoint slides are also available for each chapter. Additional material is available from the CRC Web site: http://www.crcpress.com/product/isbn/9781439862636.

# Acknowledgments

This book could not have been completed without the help of several key people. First, I would like to thank Dr. Jonathan Li for putting the book together in LaTex and for also giving some useful feedback concerning various parts of the book. In addition, he also combed through the book with me to clean up and improve the presentation. Second, Dexter Wu helped greatly with the coding of the various algorithms in the book in MATLAB® and provided solutions to the computational problems in the book. His diligence and enthusiasm in helping with the book was immensely helpful and greatly appreciated. Gina Kim provided help with many of the figures in the book and it was a joy to work with her on this project. Finally, I would like to also thank my editor Cindy Carelli for giving me the opportunity to write this book and for her patience and understanding of the several delays that occurred in the development of this work.

# *About the Author*

Roy H. Kwon is currently associate professor in the Department of Mechanical and Industrial Engineering at the University of Toronto, St. George Campus. Also, he is a member of the faculty in the Masters of Mathematical Finance (MMF) Program at the University of Toronto. He received his Ph.D. from the University of Pennsylvania in operations research from the Department of Electrical and Systems Engineering in 2002. Dr. Kwon has published articles in such journals as *Management Science, Naval Research Logistics*, the *European Journal of Operational Research*, and *Operations Research Letters* among others. In addition, he has worked and consulted in the use of operations research (optimization) for the military, financial, and service sectors.

# *Contents*

# 1

# *Linear Programming*

## 1.1 Introduction

Linear programming (LP) is the problem of optimizing (maximizing or minimizing) a linear function subject to linear constraints. A wide variety of practical problems, from nutrition, transportation, production planning, finance, and many more areas can be modeled as linear programs. We begin by introducing one of the earliest examples, the diet problem, and then give some additional applications in the areas of production management, transportation, finance, and personnel scheduling. Some of these examples are not initially linear programs, but are amenable to being transformed into LPs and techniques for conversion are illustrated. The embedded assumptions behind linear optimization problems are discussed. A definition of a standard form of a linear optimization problem is given and techniques for converting an LP into standard form are illustrated.

### 1.1.1 The Diet Problem

Due to a limited budget you would like to find the most economical mix of food items subject to providing sufficient daily nutrition. The available food items are peanut butter, bananas, and chocolate, and the cost per serving is 20 cents, 10 cents, and 15 cents, respectively. The amount of nutrients of fat, carbohydrates, and protein per serving of peanut butter is 130 grams, 51.6 grams, and 64.7 grams. For bananas, the amounts of these nutrients per serving are 1 gram, 51 grams, and 2 grams, whereas for chocolate the amounts per serving are 12 grams, 22 grams, and 2 grams. Suppose it is decided by a dietician that you need at least 35 grams of fat, at least 130 grams of carbohydrate, and at least 76 grams of protein daily. Then, a combination of these three food items should provide you with sufficient amounts of each nutrient for the day. The problem of determining the least cost combination of the food items that provide a sufficient amount of nutrients can be represented as a problem of the following form:

| | |
|---|---|
| *minimize* | cost of food items used |
| *subject to* | food items must provide enough fat |
| | food items provide must enough carbohydrates |
| | food items provide must enough protein |

This form of the problem reveals two major components, i.e., (1) there is a goal or objective (minimize cost of food items used) and (2) a set of constraints that represent the requirements for the problem (food items must provide enough nutrition). A linear programming problem will exhibit the same form, but with mathematical representations of the components of the problem. In general, a linear program will consist of decision variables, an objective, and a set of constraints.

To illustrate the formulation of the diet problem as a linear program, let

$$x_{pb} = \text{servings of peanut butter}$$
$$x_b = \text{servings of bananas}$$
$$x_c = \text{servings of chocolate.}$$

These variables represent the quantities of each food item to be used and should be non-negative, i.e., $x_{pb} \geq 0$, $x_b \geq 0$, and $x_c \geq 0$, and are called decision variables since they must be determined.

The total cost associated with any particular combination of food items $x_{pb}$, $x_b$, and $x_c$ is

$$.20x_{pb} + .10x_b + .15x_c, \tag{1.1}$$

which says that the total cost of food items used is the sum of the costs incurred from the use of each food item.

To ensure that the mix of the three foods will have enough fat one can impose the following linear inequality constraint

$$130x_{pb} + 1x_b + 12x_c \geq 35, \tag{1.2}$$

which expresses that the total amount of fat from the three food items should be at least the required minimum of 35 grams. Similarly, one can impose the inequality

$$51.6x_{pb} + 51x_b + 22x_c \geq 130 \tag{1.3}$$

to ensure that a combination of food items will have enough carbohydrates and finally impose the constraint

$$64.7x_{pb} + 2x_b + 2x_c \geq 76 \tag{1.4}$$

to ensure enough protein.

Then, the diet problem linear program can be expressed as

$$\begin{array}{ll} \textit{minimize} & .20x_{pb} + .10x_b + .15x_c \\ \textit{subject to} & 130x_{pb} + 1x_b + 12x_c \geq 35 \\ & 51.6x_{pb} + 51x_b + 22x_c \geq 130 \\ & 64.7x_{pb} + 2x_b + 2x_c \geq 76 \\ & x_{pb}, x_b, x_c \geq 0. \end{array} \tag{1.5}$$

The problem can be interpreted as the problem of determining non-negative values of $x_{pb}$, $x_b$, and $x_c$ so as to minimize the (cost) function $.20x_{pb} + .10x_b + .15x_c$ subject to meeting the nutritional requirements as embodied in the constraints (1.2)–(1.4). The function $.20x_{pb} + .10x_b + .15x_c$ is called the objective function, and the objective is the minimization of this function. It is important to observe that the objective function and the left-hand sides of the constraints are all linear, i.e., all variables are taken to the power of 1. The diet problem highlights the major components of a linear programming problem where an LP problem consists of decision variables, an objective (optimize a linear objective function), linear constraints, and possibly some sign restrictions on variables. The diet problem can be generalized to where there are $n$ types of food items and $m$ nutritional requirements, where $x_i$ represents the number of servings of food item $i$ $(i = 1, ..., n)$, $c_i$ is the cost of one serving of food item $i$ $(i = 1, , , .n)$, $a_{ij}$ is the amount of nutrient $i$ in one serving of food item $j$ $(j = 1, ..., m)$, and $b_j$ is the minimum amount of nutrient $j$ required $(j = 1, ..., m)$. Then, the general formulation of the diet problem can be written as

$$\begin{array}{ll} \text{minimize} & c_1x_1 + c_2x_2 + \cdots + c_nx_n \\ \text{subject to} & a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n \geq b_1 \\ & a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n \geq b_2 \\ & \vdots \\ & a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n \geq b_m \\ & x_1, x_2, ..., x_n \geq 0. \end{array}$$

In matrix form, the general diet problem can be represented as

$$\begin{array}{ll} \text{minimize} & c^Tx \\ \text{subject to} & Ax \geq b \\ & x \geq 0 \end{array}$$

where

$$x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \text{ is the vector of food items,}$$

$$A = [a_{ij}] = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ & & \ddots & \\ a_{m1} & a_{m2} & & a_{mn} \end{bmatrix}$$
$$= \begin{bmatrix} a_1^T \\ \vdots \\ a_m^T \end{bmatrix}$$

is a matrix of dimension $m \times n$ and $a_j^T$ is an $n$-dimensional row vector that represents the $jth$ row of $A$ i.e. the total amount of nutrient $j$ that would be obtained from the amount of food items given by $x$,

$$c = \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_n \end{bmatrix} \text{ is the vector of costs of food items,}$$

and

$$b = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix} \text{ is the vector of nutrition requirements.}$$

For example, for the diet problem (1.5) we have $n = 3$ and $m = 3$ with

$$A = \begin{bmatrix} 130 & 1 & 12 \\ 51.6 & 51 & 22 \\ 64.7 & 2 & 2 \end{bmatrix}, c = \begin{bmatrix} .20 \\ .10 \\ .15 \end{bmatrix}, x = \begin{bmatrix} x_{pb} \\ x_b \\ x_c \end{bmatrix}, \text{ and } b = \begin{bmatrix} 35 \\ 130 \\ 76 \end{bmatrix}.$$

The objective function is $c^T x = \begin{bmatrix} .20 & .10 & .15 \end{bmatrix} \cdot \begin{bmatrix} x_{pb} \\ x_b \\ x_c \end{bmatrix} = .20x_{pb} + .10x_b + .15x_c$.

(Note: The $\cdot$ indicates the dot product.)

The first constraint is

$$a_1^T x = \begin{bmatrix} 130 & 1 & 12 \end{bmatrix} \cdot \begin{bmatrix} x_{pb} \\ x_b \\ x_c \end{bmatrix} = 130x_{pb} + 1x_b + 12x_c \geq b_1 = 35.$$

The second constraint is

$$a_2^T x = \begin{bmatrix} 51.6 & 51 & 22 \end{bmatrix} \cdot \begin{bmatrix} x_{pb} \\ x_b \\ x_c \end{bmatrix} = 51.6x_{pb} + 51x_b + 22x_c \geq b_2 = 130.$$

The third constraint is

$$a_3^T x = \begin{bmatrix} 64.7 & 2 & 2 \end{bmatrix} \cdot \begin{bmatrix} x_{pb} \\ x_b \\ x_c \end{bmatrix} = 64.7x_{pb} + 2x_b + 2x_c \geq b_3 = 76.$$

### 1.1.2 Embedded Assumptions

It is important to realize the assumptions behind a linear programming problem. In particular, an LP model is characterized by proportionality, divisibility, additivity, and certainty.

1. *Proportionality* means that the contribution toward the objective function and constraints of decisions are directly proportional to its values, i.e., there are no economies of scale such as quantity-based discounts. For example, in the diet problem, every unit (serving) of peanut butter $x_{pb}$ will contribute .20 towards the overall cost and 130 grams of fat toward the fat nutrient requirement. In particular, this means that decision variables in an LP are raised to the first power only. So terms of the form $c_i x_i^{1/2}$ or $c_i x_i^2$ (where $c_i$ is a coefficient and $x_i$ a decision variable) are not permitted in an LP.

2. *Divisibility* means that the decision variables can take on any real number. For example, the amount of peanut butter in the diet problem can be less than one serving, like 35% of a serving, $x_{pb} = .35$, or in general, some fractional amount like $x_{pb} = 1.7$ servings.

3. *Additivity* means that the contribution of a decision variable toward the objective or constraints does not depend on other decision variables. In other words, the total contribution is the sum of individual contributions of each decision variable. For example, the contribution of every serving of peanut butter in the diet problem toward the overall fat requirement of 35 grams is 130 grams, independent of the amount of servings of the other food items.

4. *Certainty* means that 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. For example, it is assumed in the diet model that the cost of one serving of bananas is 10 cents and that one serving of peanut butter provides 130 grams of fat, and these values are assumed as correct or valid for the model.

The embedded assumptions may seem to be overly restrictive, but in fact a wide range of problems can be modeled as linear programs. Recent advances in optimization technology have considered the relaxation of the certainty assumption. In particular, developments in stochastic programming and robust optimization have enabled the incorporation of uncertainty of coefficients in linear programming models. These exciting topics will be covered in Chapter 8.

## 1.2 General Linear Programming Problems

The diet problem is only one possible form of a linear optimization problem. It is possible to have an LP problem that maximizes an objective function instead of minimizing and a constraint may be of the form $a^T x \geq b$ (greater than or equal inequality), $a^T x \leq b$ (less than or equal inequality), or $a^T x = b$ (equality). Furthermore, variables may be non-negative, non-positive, or unrestricted (i.e., the value can be negative, positive, or zero). For example, the following is a linear programming problem with equality and inequality constraints as well as variables that are non-negative, non-positive, and unrestricted.

$$\begin{aligned} \textit{maximize} \quad & 5x_1 + x_2 - 3x_3 \\ \textit{subject to} \quad & x_1 + x_2 \leq 6 \\ & x_2 + x_3 \geq 7 \\ & x_1 - x_3 = 2 \\ & x_1 \geq 0 \\ & x_2 \leq 0 \\ & x_3 \ \textit{unrestricted}. \end{aligned} \tag{1.6}$$

In general, a linear programming problem can be represented in the form

$$\begin{array}{lll} \textit{minimize or maximize} & c^T x & \\ \textit{subject to} & a_i^T x \leq b_i & i \in L \\ & a_i^T x \geq b_i & i \in G \\ & a_i^T x = b_i & i \in E \\ & x_j \geq 0 & j \in NN \\ & x_j \leq 0 & j \in NP. \end{array}$$

where $L, G$, and $E$ are index sets of constraints that are of the less than or equal to type, greater than or equal to type, and of equality type, respectively. $NN$ ($NP$) is an index set of variables that are non-negative (non-positive). All other variables are assumed to be unrestricted.

Furthermore, an equality constraint $a^T x = b$ can be represented as the two inequalities $a^T x \leq b$ and $a^T x \geq b$, and an inequality constraint of the form $a^T x \geq b$ can be written as a less than or equal type by multiplying both sides by $-1$ to get $-a^T x \leq -b$. Thus, the LP (1.6) is equivalent to the following

$$\begin{array}{ll} \textit{maximize} & 5x_1 + x_2 - 2x_3 \\ \textit{subject to} & x_1 + x_2 \leq 6 \\ & -x_2 - x_3 \leq -7 \\ & x_1 - x_3 \leq 2 \\ & -x_1 + x_3 \leq -2 \\ & x_1 \geq 0 \\ & x_2 \leq 0 \\ & x_3 \text{ unrestricted.} \end{array}$$

### 1.2.1 Standard Form of a Linear Program

In this section, we define the *standard form* of a linear program. A linear programming problem is said to be in standard form if (1) the objective is to minimize, (2) all constraints are of the equality type, and (3) all variables are non-negative. The following LP is in standard form:

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax = b \\ & x \geq 0. \end{array}$$

The standard form of a linear programming problem is important because some important algorithms that solve linear programs e.g. the simplex method require linear programs to be in standard form. This requirement is not too restrictive since any linear program can be converted into an equivalent linear program in standard form. The following conversion rules can be used to convert an LP in standard form.

1. *Converting unrestricted variables*

If a decision variable $x$ is initially defined to be unrestricted, i.e., the variable can take on any real number regardless of sign, then $x$ can be expressed as the difference between two non-negative numbers $x^+ \geq 0$ and $x^- \geq 0$ so that $x = x^+ - x^-$. For example, if $x = -5$, then $x^+ = 0$ and $x^- = 5$.

2. *Converting inequality constraints*

If a constraint $i$ is initially of the form $a_{i1}x_1 + a_{i2}x_2 + \cdots + a_{in}x_n \leq b_i$, then a non-negative slack variable $s_i$ can be added to the left-hand side of the constraint to get $a_{i1}x_1 + a_{i2}x_2 + \cdots + a_{in}x_n + s_i = b_i$ where $s_i \geq 0$.

If a constraint $i$ is initially of the form $a_{i1}x_1 + a_{i2}x_2 + \cdots + a_{in}x_n \geq b_i$, then a non-negative surplus variable $s_i$ can be subtracted from the left-hand side of the constraint to get $g_i(x) = a_{i1}x_1 + a_{i2}x_2 + \cdots + a_{in}x_n - s_i = b_i$ where $s_i \geq 0$.

3. *Converting maximization to minimization*

It follows that since *maximize* $c^T x = -$*minimize* $-c^T x$, any maximization problem can be converted to an equivalent minimization problem by minimizing the negated terms in the original objective function. It is common to omit the outer negation in formulations since it will not affect the optimization.

**Example 1.1**

Convert the LP

$$\begin{array}{ll} \textit{maximize} & 5x_1 - 4x_2 + 6x_3 \\ \textit{subject to} & -x_1 + x_2 \leq -7 \\ & 2x_2 - x_3 \geq 2 \\ & x_1 + 2x_3 = 7 \\ & x_1 \geq 0, x_2 \geq 0, x_3 \text{ unrestricted} \end{array}$$

into standard form.

**Solution**: The variable $x_3$ is unrestricted, so let $x_3 = x_3^+ - x_3^-$ where $x_3^+ \geq 0$ and $x_3^- \geq 0$. Then, for every occurrence of $x_3$ in the objective function and constraints, replace with $x_3^+ - x_3^-$. Next, negate the terms of the objective function to get $-5x_1 + 4x_2 - 6x_3^+ + 6x_3^-$ and then the new objective is to *minimize* $-5x_1 + 4x_2 - 6x_3^+ + 6x_3^-$. Add a slack variable $s_1$ to the left-hand side of the first constraint to get $-x_1 + x_2 + s_1 = -7$ and a surplus variable $s_2$ to the second to get $2x_2 - x_3^+ + x_3^- - s_2 = 2$. The third constraint is already in equality form. Then, the standard form of the LP is

$$\begin{array}{llllll} \textit{minimize} & -5x_1 + 4x_2 - 6x_3^+ + 6x_3^- & & & & \\ \textit{subject to} & -x_1 + \quad x_2 + & & s_1 & & = -7 \\ & \qquad 2x_2 - x_3^+ + x_3^- - & & & s_2 & = 2 \\ & x_1 + \qquad\quad 2x_3^+ - 2x_3^- & & & & = 7 \\ & x_1 \geq 0, x_2 \geq 0, x_3^+ \geq 0, x_3^- \geq 0, s_1 \geq 0, s_2 \geq 0. & & & & \end{array}$$

## 1.2.2 Linear Programming Terminology

We discuss some terminology for linear programming problems, and without loss of generality, assume that a LP is in standard form, i.e.,

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax = b \\ & x \geq 0. \end{array}$$

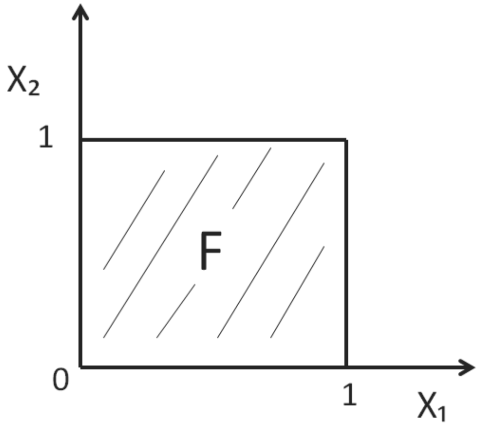

**FIGURE 1.1**
Bounded feasible set.

The matrix $A$ is called the *constraint matrix* and has dimension $m \times n$. The vector $c$ is called the *cost vector* with dimension $n \times 1$, and $b$ is called the *right-hand side vector* which is a column vector of dimension $m \times 1$. The vector $x$ is called the *decision vector* and has dimension $n \times 1$.

The set $F = \{x \in R^n | Ax = b, x \geq 0\}$ is called the *feasible set* of the linear programming problem. A vector $x \in R^n$ is said to be *feasible* for a linear program if $x \in F$, otherwise $x$ is said to be *infeasible*. A linear program is said to be *consistent* if $F \neq \varnothing$, otherwise the linear program is *inconsistent.* The feasible set is *bounded* if $\|x\| \leq M$ for all $x \in F$ for some positive constant $M$ ($\|\cdot\|$ is a norm on $x$). Intuitively, the feasible set is bounded if there is a sphere or rectangle that can completely contain the feasible set.

A vector $x^*$ is an optimal solution for a linear programming problem if $x^* \in F$ and $c^T x^* \leq c^T x$ for all $x \in F$, else $x^*$ is said to be *sub-optimal.* Also, a linear programming problem is *bounded* if $L \leq c^T x$ for all $x \in F$ for some constant $L$, else the LP is said to be unbounded. Clearly, if $F$ is bounded, then the linear program is bounded.

**Example 1.2**
Consider the following linear program (P):

$$\begin{array}{lllr} \textit{minimize} & -x_1 - x_2 & & \\ \textit{subject to} & x_1 & \leq 1 & (1.7) \\ & \quad\quad x_2 & \leq 1 & \\ & x_1 \geq 0, x_2 \geq 0. & & \end{array}$$

The feasible set is $F = \{x = (x_1, x_2)^T \in R^2 |\ x_1 \leq 1, x_2 \leq 1, x_1 \geq 0, x_2 \geq 0\}$. Converting to standard form by adding slack variables, the linear program becomes

$$\begin{array}{lllllll} \textit{minimize} & -x_1 - x_2 & & & & & \\ \textit{subject to} & x_1 & & + & x_3 & & = 1 \\ & & x_2 & + & & x_4 & = 1 \\ & \multicolumn{6}{l}{x_1 \geq 0, x_2 \geq 0, x_3 \geq 0, x_4 \geq 0.} \end{array} \tag{1.8}$$

The feasible set for (1.8) is

$$\bar{F} = \{x = (x_1, x_2, x_3, x_4)^T \in R^4 |\ x_1 + \ x_3 = 1, x_2 + \\ x_4 = 1, x_1 \geq 0, x_2 \geq 0, x_3 \geq 0, x_4 \geq 0\}.$$

Observe that the linear program (1.7) is consistent since there is at least one $x$ in $F$, e.g., $x = \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix}$ and the feasible set is bounded since for any vector $x \in F$ we have $\|x\| \leq 1$ and so (1.7) is bounded. Therefore, the linear program (1.8) is also consistent and bounded with $\bar{F}$ bounded. A graph of the feasible set $F$ is given in Figure 1.1.

The optimal solution in this example is $x^* = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \in F$, and the corresponding optimal objective function value is $-2$. Chapter 3 will discuss methods for generation and verification of optimal solutions.

**Example 1.3**
Consider the linear program

$$\begin{array}{lllll} \textit{minimize} & -x_1 - x_2 & & & \\ \textit{subject to} & & & & \\ & -x_1 & +x_3 & & = 1 \\ & \quad x_2 & & +x_4 & = 1 \\ & \multicolumn{4}{l}{x_1 \geq 0, x_2 \geq 0, x_3 \geq 0, x_4 \geq 0.} \end{array}$$

The feasible region in two dimensions is equivalent to $F = \{x = (x_1, x_2)^T \in R^2 |\ -x_1 \leq 1, x_2 \leq 1, x_1 \geq 0, x_2 \geq 0\}$ see Figure 1.2. In this case, the feasible region is not bounded as the graph extends infinitely to the right on the x-axis. In fact, for the sequence of vectors $x^{(k)} = \begin{bmatrix} k \\ 1 \end{bmatrix}$, $k = 1, 2, 3, \ldots$ , the objective function value $-k - 1 \rightarrow -\infty$ as $k \rightarrow \infty$ and so the LP is unbounded.

## 1.3 More Linear Programming Examples

This section covers some additional examples to highlight the broad range of problems that can be formulated as linear programs. Modeling a problem as a

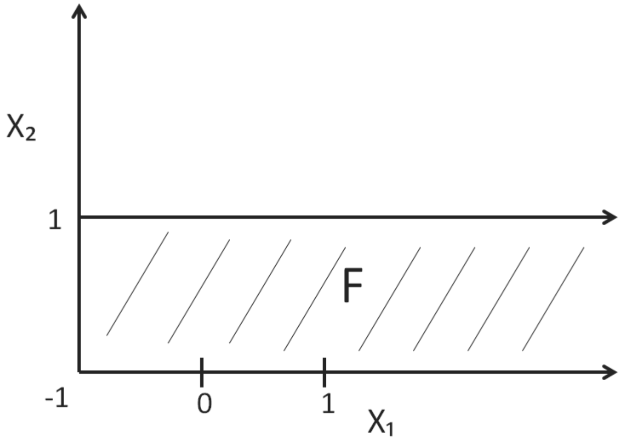

**FIGURE 1.2**
Unbounded feasible set.

linear program is an art and there is no unique way to formulate problems, but the basic requirements are the need to define decision variables, the objective, and constraints.

**Example 1.4 ( Production Planning )**

Consider a company that produces $n$ different products. Each product uses $m$ different resources. Suppose that resources are limited and the company has only $b_i$ units of resource $i$ available for each $i = 1, ..., m$. Further, each product $j$ requires $a_{ij}$ units of resource $i$ for production. Each unit of product $j$ made generates a revenue of $p_j$ dollars. The company wishes to find a production plan, i.e., the quantity of each product to produce, that maximizes revenue.

The problem can be formulated as a linear program. The first step is to define the decision variables. Let $x_j$ be a decision variable that represents the amount of product $j$ produced by the company. Suppose that fractional amounts of a product are allowed and this amount should be non-negative since a negative value for $x_j$ is meaningless. Then, a production plan is represented by the vector $x = \begin{bmatrix} x_1 & \cdots & x_n \end{bmatrix}^T$. The contribution toward revenue from the production of $x_j$ units of product $j$ is $p_j x_j$ and so the total revenue from a production plan $x$ is then $p_1x_1 + p_2x_2 + \cdots + p_nx = \sum_{j=1}^{n} p_j x_j$. The contribution toward using resource $i$ from the production of $x_j$ units of product $j$ is $a_{ij}x_j$, and so the total consumption of resource $i$ by production plan $x$ is $a_{i1}x_1 + a_{i2}x_2 + \cdots + a_{in}x_j = \sum_{j=1}^{n} a_{ij}x_j$ and this quantity can not exceed $b_i$. Since total revenue is to be maximized and resources limitations must be observed over all resources, then the LP is

$$\begin{array}{ll} \text{maximize} & \sum_{j=1}^{n} p_j x_j \\ \text{subject to} & \sum_{j=1}^{n} a_{ij} x_j \leq b_i \quad i = 1, ..., m \\ & x_j \geq 0 \quad j = 1, ..., n. \end{array}$$

**Example 1.5 (Multi-period Production Planning)**

In the previous example, the planning horizon was assumed to be one period and it was implicitly assumed that the optimal production plan generated from the model would be able to be entirely sold for the single period in order to achieve the maximum revenue. That is, the model assumed that production would occur just once and did not incorporate any consideration of demand levels and future production. We now consider a multi-period production model where production decisions are made for more than one period. Consider a single product that has demand in number of units over the next year as follows:

| Fall | Winter | Spring | Summer |
|---|---|---|---|
| 30 | 40 | 10 | 20 |

We wish to meet all demand for each period and allow excess production for a period so that it may be carried over to meet demand at future time periods. However, there will be a unit holding cost of $10 for inventory at the end of each period. Assume that there is 5 units of inventory at the start of Fall, and there is to be no inventory at the end of the Summer period.

To formulate the multi-period production model, there are some additional constructs that must be developed to capture the dynamics of production in a multiple period setting. First, in addition to decision variables that give the amount of production for each time period, there needs to be another quantity that links production from one time period to the next. These are inventory variables that indicate the amount of product in excess of the demand at the end of a time period. Then, the dynamics from one time period to the next can be captured through the following inventory balance constraint

*current inventory + production for current period = amount of product used to meet current demand + inventory for next period.*

Now, let $x_t$ = number of units of product produced in period $t$, $i_t$ = units of inventory at the end of period $t$ where Fall, Winter, Spring, and Summer correspond to periods $t = 1, 2, 3$, and, 4, respectively. Also $t = 0$ will refer to the start of Fall and $d_t$ = demand for period $t$.

Thus, the inventory balance constraints take the form $i_{t-1} + x_t = d_t + i_t$ for $t = 1, 2, 3, 4$. The objective function is to minimize total inventory costs $10i_1 + 10i_2 + 10i_3 + 10i_4$. Then, the multi-period production model is

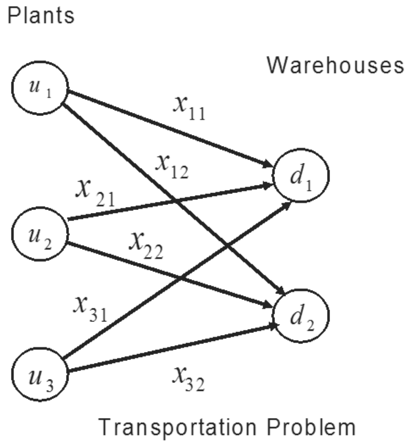

**FIGURE 1.3**
Transportation problem.

$$\begin{array}{ll} \textit{minimize} & 10i_1 + 10i_2 + 10i_3 + 10i_4 \\ \textit{subject to} & i_0 + x_1 = 30 + i_1 \\ & i_1 + x_2 = 40 + i_2 \\ & i_2 + x_3 = 10 + i_3 \\ & i_3 + x_4 = 20 + i_4 \\ & i_0 = 5 \\ & i_4 = 0 \\ & i_0, i_1, i_2, i_3, i_4, x_1, x_2, x_3, x_4 \geq 0. \end{array}$$

**Example 1.6 (Transportation Problem)**

Reactiveeno Inc. is a company that produces a special type of facial cream that aids in reducing acne. This product is manufactured in $n$ plants across North America. Every month the facial cream product is shipped from the $n$ plants to $m$ warehouses. Plant $i$ has a supply of $u_i$ units of the product. Warehouse $j$ has a demand of $d_j$ units. The cost of shipping one unit from plant $i$ to warehouse $j$ is $c_{ij}$. The problem of finding the least-cost shipping pattern from plants to warehouses can be formulated as an LP.

Let $x_{ij}$ be the number of units of product shipped from plant $i$ to warehouse $j$. So the objective is to minimize the cost of shipping over all possible plant warehouse pairs (see Figure 1.3). There are two classes of constraints. One class of constraints must ensure that the total amount shipped from a plant $i$ to warehouses, i.e., $\sum_{j=1}^{m} x_{ij}$ does not exceed the supply $u_i$ of plant $i$, whereas the other class of constraints ensures that the demand $d_j$ of a warehouse $j$ is met from the total amount $\sum_{i=1}^{n} x_{ij}$ shipped by plants. Then, the transportation problem can be formulated as the following LP.

$$\begin{array}{lll} \textit{minimize} & \sum_{i=1}^{n}\sum_{j=1}^{m} c_{ij}x_{ij} & \\ \textit{subject to} & \sum_{j=1}^{m} x_{ij} \leq u_i & i = 1, ..., n \\ & \sum_{i=1}^{n} x_{ij} \geq d_j & j = 1, ..., m \\ & x_{ij} \geq 0 & i = 1, ..., n,\ j = 1, ..., m \end{array}$$

**Example 1.7** (**The Assignment Problem**)

A special case of the transportation problem above is when there are as many plants as warehouses, i.e., $m = n$, and each warehouse demands exactly one unit of the product, and each plant produces only one unit of the product, i.e., $d_i = u_i = 1$. Then, the model takes the following form

$$\begin{array}{lll} \textit{minimize} & \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}x_{ij} & \\ \textit{subject to} & \sum_{j=1}^{n} x_{ij} = 1 & i = 1, ..., n \\ & \sum_{i=1}^{n} x_{ij} = 1 & j = 1, ..., n \\ & x_{ij} \geq 0 & i = 1, ..., n,\ j = 1, ..., n. \end{array}$$

This special case is known as the assignment problem and has the interpretation of matching persons with jobs so that each person gets one job and each job gets one person. $c_{ij}$ in this context represents the cost of assigning person $i$ to job $j$. This quantity can reflect the different skill levels of workers.

**Example 1.8** (**Workforce Scheduling**)

You are the human resources manager of a company and one of your major duties is to schedule workers for each day of the week. Each day of the week must have a required number of workers as given in the following table.

| Day of Week | Required Number of Workers |
|---|---|
| Monday | 25 |
| Tuesday | 25 |
| Wednesday | 22 |
| Thursday | 21 |
| Friday | 23 |
| Saturday | 20 |
| Sunday | 18 |

Each worker must work 5 consecutive days and then will have the next two days off. You wish to minimize the total number of workers scheduled for the week subject to providing each day with the required number of workers. The challenging aspect of this formulation resides in defining the decision variables. The variables should be defined so that it facilitates the expression of the constraints that impose a 5 consecutive day work schedule for workers. One such possibility is to let $x_j$ = number of workers that start on day $j$ where $j = 1$ corresponds to Monday, $j = 2$ corresponds to Tuesday, etc., then, the model is

$$
\begin{array}{lll}
\textit{minimize} & x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7 & \\
\textit{subject to} & x_1 + \qquad\qquad\quad x_4 + x_5 + x_6 + x_7 & \geq 25 \\
 & x_1 + x_2 + \qquad\qquad\quad x_5 + x_6 + x_7 & \geq 25 \\
 & x_1 + x_2 + x_3 \qquad\qquad\quad + x_6 + x_7 & \geq 22 \\
 & x_1 + x_2 + x_3 + x_4 \qquad\qquad\quad + x_7 & \geq 21 \\
 & x_1 + x_2 + x_3 + x_4 + x_5 & \geq 23 \\
 & \qquad x_2 + x_3 + x_4 + x_5 + x_6 & \geq 20 \\
 & \qquad\qquad x_3 + x_4 + x_5 + x_6 + x_7 & \geq 18 \\
 & x_i \geq 0 \text{ and integer} & i = 1, ..., 7.
\end{array}
$$

The *jth* constraint ensures that there will be enough workers for day $j$ by ensuring that there are enough workers that start from those days of the week for which a worker will be working on day $j$. For example, the first constraint ensures that there are enough workers for Monday by ensuring that there are enough workers that *start* work on Monday, Thursday, Friday, Saturday, and Sunday, so that there will be at least 25 workers on Monday. Observe that the coefficients of the variables $x_2$ and $x_3$ are zero in the first constraint and hence these variables do not appear in the first constraint since any worker starting on Tuesday or Wednesday and working 5 consecutive days will not be working on Monday. Note that there is an integer value requirement on the variables in addition to the non-negativity restriction since it is unreasonable to have a fractional value of a worker! So the model above violates the divisibility assumption of linear programming. The model is in fact what is called an *integer program* due to the integrality restriction of the decision variables. Otherwise, the model is very close to a linear program since the objective and constraints are linear.

**Example 1.9 (Bond Portfolio Optimization)**
Suppose that a bank has the following liability schedule

| Year 1 | Year 2 | Year 3 |
|---|---|---|
| \$12,000 | \$18,000 | \$20,000 |

That is, the bank needs to pay \$12,000 at the end of the first year, \$18,000 at the end of the second year, and \$20,000 at the end of the third year. Bonds are securities that are sold by agencies, such as corporations or governments, that entitle the buyer to periodic interest (coupon) payments and the payment of the principle (face value) at some time in the future (maturity).

The bank wishes to use the three bonds below to form a portfolio (a collection of bonds) today to hold until all bonds have matured and that will generate the required cash to meet the liabilities. All bonds have a face value of a \$100 and the coupons are annual (with one coupon per year). For example, one unit of Bond 2 costs \$99 now and the holder will receive \$3.50 after 1 year and then \$3.50 plus the face value of \$100 at the end of the second year which is the maturity of Bond 2.

| Bond | 1 | 2 | 3 |
|---|---|---|---|
| Price | \$102 | \$99 | \$98 |
| Coupon | \$5 | \$3.5 | \$3.5 |
| Maturity year | 1 | 2 | 3 |

The bank wishes to purchase Bonds 1, 2, and 3 in amounts whose total cash flow will offset the liabilities. Assume that fractional amounts of each bond are permitted. A linear programming model can be formulated to find the lowest-cost set of bonds (i.e., portfolio) consisting of Bonds 1, 2, and 3 above that will meet the liabilities. Let the $x_i$ = amount of bond $i$ purchased. Then, the problem can be modeled as follows

$$\begin{array}{lrl}
\textit{minimize} & 102x_1 + 99x_2 + 98x_3 & \\
\textit{subject to} & 105x_1 + 3.5x_2 + 3.5x_3 & \geq 12000 \\
 & 103.5x_2 + 3.5x_3 & \geq 18000 \\
 & 103.5x_3 & \geq 20000 \\
 & x_1, x_2, x_3 \geq 0. &
\end{array}$$

The objective is to minimize the cost of a portfolio and each constraint ensures that that cash flow generated by the bonds for a given time period is sufficient to match the liability for that period. For example, in the first constraint each unit of Bond 1 purchased will generate \$5 from the coupon plus the \$100 face value (since bonds of type 1 mature at the end of year 1) so the total cash contribution from Bond 1 is \$$105x_1$, the total cash flow from Bond 2 is only \$$3.5x_2$ since these bonds do not mature until the end of year 2 but only payout \$3.5 per unit for the coupon at the end of year 1. The total cash flow from bonds of type 3 is also \$$3.5x_3$ at the end of the first year. Note that in constraint 2 there is no term involving bonds of type 1 since they have already matured after one year and can no longer generate cash flow.

### 1.3.1 Converting Minimization Problems with Absolute Value

Consider an optimization problem of the following form

$$\begin{array}{lll}
\textit{minimize} & c_1|x_1| + c_2|x_2| + \cdots + c_n|x_n| & \\
\textit{subject to} & x_i \text{ unrestricted} & i = 1, ...n
\end{array}$$

where $c_i > 0$ for all $i = 1, ...n$. The problem in the form above is not a linear program since the absolute value terms in the objective function are not linear. However, the problem can be converted into an equivalent linear program through the following transformation. Let

$$|x_i| = x_i^+ + x_i^- \text{ and } x_i = x_i^+ - x_i^- \text{ where } x_i^+, x_i^- \geq 0.$$

Since the objective is to minimize and $c_i > 0$, then $x_i^+ \times x_i^- = 0$ will hold at optimality, this will ensure that the transformation is equivalent to the absolute value of $x_i$ where if $x_i \geq 0$, then $|x_i| = x_i^+ = x_i$ and $x_i^- = 0$ else, $|x_i| = x_i^- = -x_i$ and $x_i^+ = 0$.

**Example 1.10**

The optimization problem above can be transformed to an LP by replacing each occurrence of $|x_i|$ with $x_i^+ + x_i^-$ and by adding $x_i = x_i^+ - x_i^-$ as a constraint along with the restrictions $x_i^+ \geq 0$ and $x_i^- \geq 0$. The model then becomes

$$\begin{array}{ll} \textit{minimize} & c_1(x_1^+ + x_1^-) + c_2(x_2^+ + x_2^-) + \cdots + c_n(x_n^+ + x_n^-) \\ \textit{subject to} & x_1 = \ x_1^+ - x_1^- \\ & x_2 = \ x_2^+ - x_2^- \\ & \qquad \vdots \\ & x_n = \ x_n^+ - x_n^- \\ & x_1^+ \geq 0, x_1^- \geq 0, x_2^+ \geq 0, x_2^- \geq 0, ..., x_n^+ \geq 0, x_n^- \geq 0. \end{array}$$

**Example 1.11 (Application: Portfolio Optimization)**

Consider the problem of investing money in $n$ stocks where each stock $i$ has a random rate of return $r_i$ with an expected return of $\mu_i$. $r_i$ models the price uncertainty for stock $i$ and is often assumed to be a normal distribution. In addition, the covariance between the returns of stock $i$ and stock $j$ is $\sigma_{ij}$. Let $x_i$= the proportion of wealth invested in stock $i$. A portfolio is then represented by the vector $x = (x_1, ..., x_n)^T$. A reasonable model to use to construct a portfolio is the following model developed by Markowitz (1952), which is a one-period model where an investment is made now and held until a future point in time $T$.

$$\begin{array}{lll} \textit{minimize} & \sum_{i=1}^n \sum_{j=1}^n \sigma_{ij} x_i x_j & \\ \textit{subject to} & \sum_{i=1}^n \mu_i x_i & = R \\ & \sum_{i=1}^n x_i & = 1 \\ & \quad x_i \geq 0 & i = 1, ..., n \end{array}$$

The objective function of the model is the variance of the return of the portfolio $x$ where the variance represents the risk of the portfolio. Then, the objective is to minimize the risk (portfolio variance) subject to meeting an expected return goal of $R$ for the portfolio (first constraint) and ensuring that the budget is exhausted (second constraint). In other words, the goal is to find the portfolio among all of the portfolios that can achieve an expected return of $R$ while exhausting the budget, and that has the smallest variance among them.

However, there are several challenges in using the Markowitz model. First, observe that the model is non-linear since the terms in the objective function