

OPTIMIZATION METHODS IN FINANCE

SECOND EDITION



Gérard Cornuéjols,
Javier Peña and Reha Tütüncü

Optimization Methods in Finance

Optimization methods play a central role in financial modeling. This textbook is devoted to explaining how state-of-the-art optimization theory, algorithms, and software can be used to efficiently solve problems in computational finance. It discusses some classical mean–variance portfolio optimization models as well as more modern developments such as models for optimal trade execution and dynamic portfolio allocation with transaction costs and taxes. Chapters discussing the theory and efficient solution methods for the main classes of optimization problems alternate with chapters discussing their use in the modeling and solution of central problems in mathematical finance.

This book will be interesting and useful for students, academics, and practitioners with a background in mathematics, operations research, or financial engineering.

The second edition includes new examples and exercises as well as a more detailed discussion of mean–variance optimization, multi-period models, and additional material to highlight the relevance to finance.

G rard Cornu jols is a Professor of Operations Research at the Tepper School of Business, Carnegie Mellon University. He is a member of the National Academy of Engineering and has received numerous prizes for his research contributions in integer programming and combinatorial optimization, including the Lanchester Prize, the Fulkerson Prize, the Dantzig Prize, and the von Neumann Theory Prize.

Javier Pe a is a Professor of Operations Research at the Tepper School of Business, Carnegie Mellon University. His research explores the myriad of challenges associated with large-scale optimization models and he has published numerous articles on optimization, machine learning, financial engineering, and computational game theory. His research has been supported by grants from the National Science Foundation, including a prestigious CAREER award.

Reha T t nc  is the Chief Risk Officer at SECOR Asset Management and an adjunct professor at Carnegie Mellon University. He has previously held senior positions at Goldman Sachs Asset Management and AQR Capital Management focusing on quantitative portfolio construction, equity portfolio management, and risk management.

Optimization Methods in Finance

Second Edition

GÉRARD CORNUÉJOLS

Carnegie Mellon University, Pennsylvania

JAVIER PEÑA

Carnegie Mellon University, Pennsylvania

REHA TÜTÜNCÜ

SECOR Asset Management



CAMBRIDGE
UNIVERSITY PRESS

CAMBRIDGE

UNIVERSITY PRESS

University Printing House, Cambridge CB2 8BS, United Kingdom

One Liberty Plaza, 20th Floor, New York, NY 10006, USA

477 Williamstown Road, Port Melbourne, VIC 3207, Australia

314–321, 3rd Floor, Plot 3, Splendor Forum, Jasola District Centre, New Delhi – 110025, India

79 Anson Road, #06–04/06, Singapore 079906

Cambridge University Press is part of the University of Cambridge.

It furthers the University's mission by disseminating knowledge in the pursuit of education, learning, and research at the highest international levels of excellence.

www.cambridge.org

Information on this title: www.cambridge.org/9781107056749

DOI: [10.1017/9781107297340](https://doi.org/10.1017/9781107297340)

First edition © Gérard Cornuéjols and Reha Tütüncü 2007

Second edition © Gérard Cornuéjols, Javier Peña and Reha Tütüncü 2018

This publication is in copyright. Subject to statutory exception and to the provisions of relevant collective licensing agreements, no reproduction of any part may take place without the written permission of Cambridge University Press.

First published 2007

Second edition 2018

Printed in the United Kingdom by TJ International Ltd. Padstow Cornwall

A catalogue record for this publication is available from the British Library.

ISBN 978-1-107-05674-9 Hardback

Additional resources for this publication at www.cambridge.org/9781107056749

Cambridge University Press has no responsibility for the persistence or accuracy of URLs for external or third-party internet websites referred to in this publication and does not guarantee that any content on such websites is, or will remain, accurate or appropriate.

Contents

Preface

page xi

Part I	Introduction	1
1	Overview of Optimization Models	3
1.1	Types of Optimization Models	4
1.2	Solution to Optimization Problems	7
1.3	Financial Optimization Models	8
1.4	Notes	10
2	Linear Programming: Theory and Algorithms	11
2.1	Linear Programming	11
2.2	Graphical Interpretation of a Two-Variable Example	15
2.3	Numerical Linear Programming Solvers	16
2.4	Sensitivity Analysis	17
2.5	*Duality	20
2.6	*Optimality Conditions	23
2.7	*Algorithms for Linear Programming	24
2.8	Notes	30
2.9	Exercises	31
3	Linear Programming Models: Asset–Liability Management	35
3.1	Dedication	35
3.2	Sensitivity Analysis	38
3.3	Immunization	38
3.4	Some Practical Details about Bonds	41
3.5	Other Cash Flow Problems	44
3.6	Exercises	47
3.7	Case Study	51
4	Linear Programming Models: Arbitrage and Asset Pricing	53
4.1	Arbitrage Detection in the Foreign Exchange Market	53
4.2	The Fundamental Theorem of Asset Pricing	55
4.3	One-Period Binomial Pricing Model	56

4.4	Static Arbitrage Bounds	59
4.5	Tax Clientele Effects in Bond Portfolio Management	63
4.6	Notes	65
4.7	Exercises	65
Part II	Single-Period Models	69
5	Quadratic Programming: Theory and Algorithms	71
5.1	Quadratic Programming	71
5.2	Numerical Quadratic Programming Solvers	74
5.3	Sensitivity Analysis	75
5.4	*Duality and Optimality Conditions	76
5.5	*Algorithms	81
5.6	Applications to Machine Learning	84
5.7	Exercises	87
6	Quadratic Programming Models: Mean–Variance Optimization	90
6.1	Portfolio Return	90
6.2	Markowitz Mean–Variance (Basic Model)	91
6.3	Analytical Solutions to Basic Mean–Variance Models	95
6.4	More General Mean–Variance Models	99
6.5	Portfolio Management Relative to a Benchmark	103
6.6	Estimation of Inputs to Mean–Variance Models	106
6.7	Performance Analysis	112
6.8	Notes	115
6.9	Exercises	115
6.10	Case Studies	121
7	Sensitivity of Mean–Variance Models to Input Estimation	124
7.1	Black–Litterman Model	126
7.2	Shrinkage Estimation	129
7.3	Resampled Efficiency	131
7.4	Robust Optimization	132
7.5	Other Diversification Approaches	133
7.6	Exercises	135
8	Mixed Integer Programming: Theory and Algorithms	140
8.1	Mixed Integer Programming	140
8.2	Numerical Mixed Integer Programming Solvers	143
8.3	Relaxations and Duality	145
8.4	Algorithms for Solving Mixed Integer Programs	150
8.5	Exercises	157

9	Mixed Integer Programming Models: Portfolios with Combinatorial Constraints	161
9.1	Combinatorial Auctions	161
9.2	The Lockbox Problem	163
9.3	Constructing an Index Fund	165
9.4	Cardinality Constraints	167
9.5	Minimum Position Constraints	168
9.6	Risk-Parity Portfolios and Clustering	169
9.7	Exercises	169
9.8	Case Study	171
10	Stochastic Programming: Theory and Algorithms	173
10.1	Examples of Stochastic Optimization Models	173
10.2	Two-Stage Stochastic Optimization	174
10.3	Linear Two-Stage Stochastic Programming	175
10.4	Scenario Optimization	176
10.5	*The L-Shaped Method	177
10.6	Exercises	179
11	Stochastic Programming Models: Risk Measures	181
11.1	Risk Measures	181
11.2	A Key Property of CVaR	185
11.3	Portfolio Optimization with CVaR	186
11.4	Notes	190
11.5	Exercises	190
Part III	Multi-Period Models	195
12	Multi-Period Models: Simple Examples	197
12.1	The Kelly Criterion	197
12.2	Dynamic Portfolio Optimization	198
12.3	Execution Costs	201
12.4	Exercises	209
13	Dynamic Programming: Theory and Algorithms	212
13.1	Some Examples	212
13.2	Model of a Sequential System (Deterministic Case)	214
13.3	Bellman's Principle of Optimality	215
13.4	Linear–Quadratic Regulator	216
13.5	Sequential Decision Problem with Infinite Horizon	218
13.6	Linear–Quadratic Regulator with Infinite Horizon	219
13.7	Model of Sequential System (Stochastic Case)	221
13.8	Notes	222
13.9	Exercises	222

14	Dynamic Programming Models: Multi-Period Portfolio Optimization	225
14.1	Utility of Terminal Wealth	225
14.2	Optimal Consumption and Investment	227
14.3	Dynamic Trading with Predictable Returns and Transaction Costs	228
14.4	Dynamic Portfolio Optimization with Taxes	230
14.5	Exercises	234
15	Dynamic Programming Models: the Binomial Pricing Model	238
15.1	Binomial Lattice Model	238
15.2	Option Pricing	238
15.3	Option Pricing in Continuous Time	244
15.4	Specifying the Model Parameters	245
15.5	Exercises	246
16	Multi-Stage Stochastic Programming	248
16.1	Multi-Stage Stochastic Programming	248
16.2	Scenario Optimization	250
16.3	Scenario Generation	255
16.4	Exercises	259
17	Stochastic Programming Models: Asset–Liability Management	262
17.1	Asset–Liability Management	262
17.2	The Case of an Insurance Company	263
17.3	Option Pricing via Stochastic Programming	265
17.4	Synthetic Options	270
17.5	Exercises	273
Part IV	Other Optimization Techniques	275
18	Conic Programming: Theory and Algorithms	277
18.1	Conic Programming	277
18.2	Numerical Conic Programming Solvers	282
18.3	Duality and Optimality Conditions	282
18.4	Algorithms	284
18.5	Notes	287
18.6	Exercises	287
19	Robust Optimization	289
19.1	Uncertainty Sets	289
19.2	Different Flavors of Robustness	290
19.3	Techniques for Solving Robust Optimization Models	294
19.4	Some Robust Optimization Models in Finance	297
19.5	Notes	302
19.6	Exercises	302

20	Nonlinear Programming: Theory and Algorithms	305
20.1	Nonlinear Programming	305
20.2	Numerical Nonlinear Programming Solvers	306
20.3	Optimality Conditions	306
20.4	Algorithms	308
20.5	Estimating a Volatility Surface	315
20.6	Exercises	319
Appendices		321
Appendix	Basic Mathematical Facts	323
A.1	Matrices and Vectors	323
A.2	Convex Sets and Convex Functions	324
A.3	Calculus of Variations: the Euler Equation	325
References		327
Index		334

Preface

The use of sophisticated mathematical tools in modern finance is now commonplace. Researchers and practitioners routinely run simulations or solve differential equations to price securities, estimate risks, or determine hedging strategies. Some of the most important tools employed in these computations are optimization algorithms. Many computational finance problems ranging from asset allocation to risk management, from option pricing to model calibration, can be solved by optimization techniques. This book is devoted to explaining how to solve such problems efficiently and accurately using the state of the art in optimization models, methods, and software.

Optimization is a mature branch of applied mathematics. Typical optimization problems have the goal of allocating limited resources to alternative activities in order to maximize the total benefit obtained from these activities. Through decades of intensive and innovative research, fast and reliable algorithms and software have become available for many classes of optimization problems. Consequently, optimization is now being used as an effective management and decision-support tool in many industries, including the financial industry.

This book discusses several classes of optimization problems encountered in financial models, including linear, quadratic, integer, dynamic, stochastic, conic, and nonlinear programming. For each problem class, after introducing the relevant theory (optimality conditions, duality, etc.) and efficient solution methods, we discuss several problems of mathematical finance that can be modeled within this problem class.

The second edition includes a more detailed discussion of mean–variance optimization, multi-period models, and additional material to highlight the relevance to finance.

The book’s structure has also been clarified for the second edition; it is now organized in four main parts, each comprising several chapters. Part I guides the reader through the solution of asset liability cash flow matching using linear programming techniques, which are also used to explain asset pricing and arbitrage. Part II is devoted to single-period models. It provides a thorough treatment of mean–variance portfolio optimization models, including derivations of the one-fund and two-fund theorems and their connection to the capital asset pricing model, a discussion of linear factor models that are used extensively

in risk and portfolio management, and techniques to deal with the sensitivity of mean–variance models to parameter estimation. We discuss integer programming formulations for portfolio construction problems with cardinality constraints, and we explain how this is relevant to constructing an index fund. The final chapters of Part II present a stochastic programming approach to modeling measures of risk other than the variance, including the popular value at risk and conditional value at risk.

Part III of the book discusses multi-period models such as the iconic Kelly criterion and binomial lattice models for asset pricing as well as more elaborate and modern models for optimal trade execution, dynamic portfolio optimization with transaction costs and taxes, and asset–liability management. These applications showcase techniques from dynamic and stochastic programming.

Part IV is devoted to more advanced optimization techniques. We introduce conic programming and discuss applications such as the approximation of covariance matrices and robust portfolio optimization. The final chapter of Part IV covers one of the most general classes of optimization models, namely nonlinear programming, and applies it to volatility estimation.

This book is intended as a textbook for Master’s programs in financial engineering, finance, or computational finance. In addition, the structure of chapters, alternating between optimization methods and financial models that employ these methods, allows the book to be used as a primary or secondary text in upper-level undergraduate or introductory graduate courses in operations research, management science, and applied mathematics. A few sections are marked with a ‘*’ to indicate that the material they contain is more technical and can be safely skipped without loss of continuity.

Optimization algorithms are sophisticated tools and the relationship between their inputs and outputs is sometimes opaque. To maximize the value from using these tools and to understand how they work, users often need a significant amount of guidance and practical experience with them. This book aims to provide this guidance and serve as a reference tool for the finance practitioners who use or want to use optimization techniques.

This book has benefited from the input provided by instructors and students in courses at various institutions. We thank them for their valuable feedback and for many stimulating discussions. We would also like to thank the colleagues who provided the initial impetus for this book and colleagues who collaborated with us on various research projects that are reflected in the book. We especially thank Kathie Cameron, the late Rick Green, Raphael Hauser, John Hooker, Miroslav Karamanov, Mark Koenig, Masakazu Kojima, Vijay Krishnamurthy, Miguel Lejeune, Yanjun Li, François Margot, Ana Margarida Monteiro, Mustafa Pinar, Sebastian Pokutta, Sanjay Srivastava, Michael Trick, and Luís Vicente.

Part I

Introduction

1 Overview of Optimization Models

Optimization is the process of finding the *best* way of making decisions that satisfy a set of constraints. In mathematical terms, an optimization model is a problem of the form

$$\begin{array}{ll} \min_{\mathbf{x}} & f(\mathbf{x}) \\ \text{s.t.} & \mathbf{x} \in \mathcal{X}, \end{array} \quad (1.1)$$

where $f : \mathbb{R}^n \rightarrow \mathbb{R}$ and $\mathcal{X} \subseteq \mathbb{R}^n$.

Model (1.1) has three main components, namely the vector of *decision variables* $\mathbf{x} := [x_1 \ \cdots \ x_n]^\top \in \mathbb{R}^n$; the *objective function* $f(\mathbf{x})$; and the *constraint set* or *feasible region* \mathcal{X} . The constraint set is often expressed in terms of equalities and inequalities involving additional functions. More precisely, the constraint set \mathcal{X} is often of the form

$$\mathcal{X} = \{\mathbf{x} \in \mathbb{R}^n : g_i(\mathbf{x}) = b_i, \text{ for } i = 1, \dots, m, \text{ and } h_j(\mathbf{x}) \leq d_j, \text{ for } j = 1, \dots, p\}, \quad (1.2)$$

for some $g_i, h_j : \mathbb{R}^n \rightarrow \mathbb{R}$, $i = 1, \dots, m$, $j = 1, \dots, p$. When this is the case, the optimization problem (1.1) is usually written in the form

$$\begin{array}{ll} \min_{\mathbf{x}} & f(\mathbf{x}) \\ \text{s.t.} & g_i(\mathbf{x}) = b_i, \text{ for } i = 1, \dots, m \\ & h_j(\mathbf{x}) \leq d_j, \text{ for } j = 1, \dots, p, \end{array}$$

or in the more concise form

$$\begin{array}{ll} \min_{\mathbf{x}} & f(\mathbf{x}) \\ \text{s.t.} & \mathbf{g}(\mathbf{x}) = \mathbf{b} \\ & \mathbf{h}(\mathbf{x}) \leq \mathbf{d}. \end{array}$$

We will use the following terminology. A *feasible point* or *feasible solution* to (1.1) is a point in the constraint set \mathcal{X} . An *optimal solution* to (1.1) is a feasible point that attains the best possible objective value; that is, a point $\mathbf{x}^* \in \mathcal{X}$ such that $f(\mathbf{x}^*) \leq f(\mathbf{x})$ for all $\mathbf{x} \in \mathcal{X}$. The *optimal value* of (1.1) is the value of the objective function at an optimal solution; that is, $f(\mathbf{x}^*)$ where \mathbf{x}^* is an optimal solution to (1.1). If the feasible region \mathcal{X} is of the form (1.2) and $\mathbf{x} \in \mathcal{X}$, the *binding constraints* at \mathbf{x} are the equality constraints and those inequality constraints that hold with equality at \mathbf{x} . The term *active constraint* is also often used in lieu of “binding constraint”. The problem (1.1) is *infeasible* if $\mathcal{X} = \emptyset$. On

the other hand, (1.1) is *unbounded* if there exist $\mathbf{x}_k \in \mathcal{X}$, $k = 1, 2, \dots$, such that $f(\mathbf{x}_k) \rightarrow -\infty$.

1.1 Types of Optimization Models

For optimization models to be of practical interest, their computational tractability, that is, the ability to find the optimal solution efficiently, is a critical issue. Particular structural assumptions on the objective and constraints of the problem give rise to different classes of optimization models with various degrees of computational difficulty. We should note that the following is only a partial classification based on the current generic tractability of various types of optimization models. However, what is “tractable” in some specific context may be more nuanced. Furthermore, tractability evolves as new algorithms and technologies are developed.

Convex optimization: These are problems where the objective $f(\mathbf{x})$ is a convex function and the constraint set \mathcal{X} is a convex set. This class of optimization models is tractable most of the time. By this we mean that a user can expect any of these models to be amenable to an efficient algorithm. We will emphasize this class of optimization models throughout the book.

Mixed integer optimization: These are problems where some of the variables are restricted to take integer values. This restriction makes the constraint set \mathcal{X} non-convex. This class of optimization models is somewhat tractable a fair portion of the time. By this we mean that a model of this class may be solvable provided the user does some judicious modeling and has access to high computational power.

Stochastic and dynamic optimization: These are problems involving random and time-dependent features. This class of optimization models is tractable only in some special cases. By this we mean that, unless some specific structure and assumptions hold, a model of this class would typically be insoluble with any realistic amount of computational power at our disposal. Current research is expected to enrich the class of tractable models in this area.

The modeling of time and uncertainty is pervasive in almost every financial problem. The various types of optimization problems that we will discuss are based on how they deal with these two issues. Generally speaking, *static models* are associated with simple single-period models where the future is modeled as a single stage. By contrast, in *multi-period* models the future is modeled as a sequence, or possibly as a continuum, of stages. With regard to uncertainty, *deterministic models* are those where all the defining data are assumed to be known with certainty. By contrast, *stochastic models* are ones that incorporate probabilistic or other types of uncertainty in the data.

A good portion of the models that we will present in this book will be convex optimization models due to their favorable mathematical and computational properties. There are two special types of convex optimization problems that we will use particularly often: *linear* and *quadratic* programming, the latter being an extension of the former. These two types of optimization models will be discussed in more detail in [Chapters 2](#) and [5](#). We now present a high-level description of four major classes of optimization models: linear programming, quadratic programming, mixed integer programming, and stochastic optimization.

Linear Programming

A linear programming model is an optimization problem where the objective is a linear function and the constraint set is defined by finitely many linear equalities and linear inequalities. In other words, a linear program is a problem of the form

$$\begin{array}{ll} \min_{\mathbf{x}} & \mathbf{c}^T \mathbf{x} \\ \text{s.t.} & \mathbf{Ax} = \mathbf{b} \\ & \mathbf{Dx} \geq \mathbf{d} \end{array}$$

for some vectors $\mathbf{c} \in \mathbb{R}^n$, $\mathbf{b} \in \mathbb{R}^m$, $\mathbf{d} \in \mathbb{R}^p$ and matrices $\mathbf{A} \in \mathbb{R}^{m \times n}$, $\mathbf{D} \in \mathbb{R}^{p \times n}$.

The term *linear optimization* is sometimes used in place of linear programming. The wide popularity of linear programming is due in good part to the availability of very efficient algorithms. The two best known and most successful methods for solving linear programs are the *simplex method* and *interior-point methods*. We briefly discuss these algorithms in [Chapter 2](#).

Quadratic Programming

Quadratic programming, also known as quadratic optimization, is an extension of linear programming where the objective function includes a quadratic term. In other words, a quadratic program is a problem of the form

$$\begin{array}{ll} \min_{\mathbf{x}} & \frac{1}{2} \mathbf{x}^T \mathbf{Q} \mathbf{x} + \mathbf{c}^T \mathbf{x} \\ \text{s.t.} & \mathbf{Ax} = \mathbf{b} \\ & \mathbf{Dx} \geq \mathbf{d} \end{array}$$

for some vectors and matrices $\mathbf{Q} \in \mathbb{R}^{n \times n}$, $\mathbf{c} \in \mathbb{R}^n$, $\mathbf{b} \in \mathbb{R}^m$, $\mathbf{d} \in \mathbb{R}^p$, $\mathbf{A} \in \mathbb{R}^{m \times n}$, $\mathbf{D} \in \mathbb{R}^{p \times n}$. It is customary to assume that the matrix \mathbf{Q} is symmetric. This assumption can be made without loss of generality since

$$\mathbf{x}^T \mathbf{Q} \mathbf{x} = \mathbf{x}^T \tilde{\mathbf{Q}} \mathbf{x}$$

where $\tilde{\mathbf{Q}} = \frac{1}{2}(\mathbf{Q} + \mathbf{Q}^T)$, which is clearly a symmetric matrix.

We note that a quadratic function $\frac{1}{2} \mathbf{x}^T \mathbf{Q} \mathbf{x} + \mathbf{c}^T \mathbf{x}$ is convex if and only if the matrix \mathbf{Q} is positive semidefinite ($\mathbf{x}^T \mathbf{Q} \mathbf{x} \geq 0$ for all $\mathbf{x} \in \mathbb{R}^n$). In this case the above quadratic program is a convex optimization problem and can be solved

efficiently. The two best known methods for solving convex quadratic programs are *active-set methods* and *interior-point methods*. We briefly discuss these algorithms in [Chapter 5](#).

Mixed Integer Programming

A mixed-integer program is an optimization problem that restricts some or all of the decision variables to take integer values. In particular, a mixed integer linear programming model is a problem of the form

$$\begin{aligned} \min_{\mathbf{x}} \quad & \mathbf{c}^\top \mathbf{x} \\ \text{s.t.} \quad & \mathbf{A}\mathbf{x} = \mathbf{b} \\ & \mathbf{D}\mathbf{x} \geq \mathbf{d} \\ & x_j \in \mathbb{Z}, \quad j \in J \end{aligned}$$

for some vectors and matrices $\mathbf{c} \in \mathbb{R}^n$, $\mathbf{b} \in \mathbb{R}^m$, $\mathbf{d} \in \mathbb{R}^p$, $\mathbf{A} \in \mathbb{R}^{m \times n}$, $\mathbf{D} \in \mathbb{R}^{p \times n}$ and some $J \subseteq \{1, \dots, n\}$.

An important case occurs when the model includes *binary* variables, that is, variables that are restricted to take values 0 or 1. As we will see, the inclusion of this type of constraint increases the modeling power but comes at a cost in terms of computational tractability. It is noteworthy that the computational and algorithmic machinery for solving mixed integer programs has vastly improved during the last couple of decades. The main classes of methods for solving mixed integer programs are *branch and bound*, *cutting planes*, and a combination of these two approaches known as *branch and cut*. We briefly discuss these algorithms in [Chapter 8](#).

Stochastic Optimization

Stochastic optimization models are optimization problems that account for randomness in their objective or constraints. The following formulation illustrates a generic type of stochastic optimization problem

$$\begin{aligned} \min_{\mathbf{x}} \quad & \mathbb{E}(F(\mathbf{x}, \omega)) \\ & \mathbf{x} \in \mathcal{X}. \end{aligned}$$

In this problem the set of decisions \mathbf{x} must be made before a random outcome ω occurs. The goal is to optimize the expectation of some function that depends on both the decision vector \mathbf{x} and the random outcome ω . A variation of this formulation, that has led to important developments, is to replace the expectation by some kind of *risk measure* ϱ in the objective:

$$\begin{aligned} \min_{\mathbf{x}} \quad & \varrho(F(\mathbf{x}, \omega)) \\ & \mathbf{x} \in \mathcal{X}. \end{aligned}$$

There are numerous refinements and variants of the above two formulations. In particular, the class of *two-stage stochastic optimization with recourse* has been

widely studied in the stochastic programming community. In this setting a set of decisions \mathbf{x} must be made in stage one. Between stage one and stage two a random outcome ω occurs. At stage two we have the opportunity to make some second-stage *recourse* decisions $\mathbf{y}(\omega)$ that may depend on the random outcome ω .

The two-stage stochastic optimization problem with recourse can be formally stated as

$$\min_{\mathbf{x}} \quad f(\mathbf{x}) + \mathbb{E}[Q(\mathbf{x}, \omega)]$$

$$\mathbf{x} \in \mathcal{X}.$$

The *recourse* term $Q(\mathbf{x}, \omega)$ depends on the first-stage decisions \mathbf{x} and the random outcome ω . It is of the form

$$Q(\mathbf{x}, \omega) := \min_{\mathbf{y}(\omega)} \quad g(\mathbf{y}(\omega), \omega)$$

$$\mathbf{y}(\omega) \in \mathcal{Y}(\mathbf{x}, \omega).$$

The second-stage decisions $\mathbf{y}(\omega)$ are *adaptive* to the random outcome ω because they are made after ω is revealed. The objective function in a two-stage stochastic optimization problem contains a term for the stage-one decisions and a term for the stage-two decisions where the latter term involves an expectation over the random outcomes. The intuition of this objective function is that the stage-one decisions should be made considering what is to be expected in stage two.

The above two-stage setting generalizes to a multi-stage context where the random outcome is revealed over time and decisions are made dynamically at multiple stages and can adapt to the information revealed up to their stage.

1.2 Solution to Optimization Problems

The solution to an optimization problem can often be characterized in terms of a set of *optimality conditions*. Optimality conditions are derived from the mathematical relationship between the objective and constraints in the problem. Subsequent chapters discuss optimality conditions for various types of optimization problems. In special cases, these optimality conditions can be solved analytically and used to infer properties about the optimal solution. However, in many cases we rely on numerical solvers to obtain the solution to the optimization models.

There are numerous software vendors that provide solvers for optimization problems. Throughout this book we will illustrate examples with two popular solvers, namely Excel Solver and the MATLAB®-based optimization modeling framework CVX. Excel and MATLAB files for the examples and exercises in the book are available at:

www.andrew.cmu.edu/user/jfp/0IFbook/

Both Excel Solver and CVX enable us to solve small to medium-sized problems and are fairly easy to use. There are far more sophisticated solvers such as the

commercial solvers IBM®–ILOG® CPLEX®, Gurobi, FICO® Xpress, and the ones available via the open-source projects COIN-OR or SCIP.

Optimization problems can be formulated using modeling languages such as AMPL, GAMS, MOSEL, or OPL. The need for these modeling languages arises when the size of the formulation is large. A modeling language lets people use common notation and familiar concepts to formulate optimization models and examine solutions. Most importantly, large problems can be formulated in a compact way. Once the problem has been formulated using a modeling language, it can be solved using any number of solvers. A user can switch between solvers with a single command and select options that may improve solver performance.

1.3 Financial Optimization Models

In this book we will focus on the use of optimization models for financial problems such as portfolio management, risk management, asset and liability management, trade execution, and dynamic asset management. Optimization models are also widely used in other areas of business, science, and engineering, but this will not be the subject of our discussion.

Portfolio Management

One of the best known optimization models in finance is the portfolio selection model of Markowitz (1952). Markowitz’s mean–variance approach led to major developments in financial economics including Tobin’s mutual fund theorem (Tobin, 1958) and the capital asset pricing model of Treynor¹, Sharpe (1964), Lintner (1965), and Mossin (1966). Markowitz was awarded the Nobel Prize in Economics in 1990 for the enormous influence of his work in financial theory and practice. The gist of this model is to formalize the principle of diversification when selecting a portfolio in a universe of risky assets. As we discuss in detail in Chapter 6, Markowitz’s mean–variance model and a wide range of its variations can be stated as a quadratic programming problem of the form

$$\begin{aligned} \min_{\mathbf{x}} \quad & \frac{1}{2} \gamma \cdot \mathbf{x}^T \mathbf{V} \mathbf{x} - \boldsymbol{\mu}^T \mathbf{x} \\ & \mathbf{A} \mathbf{x} = \mathbf{b} \\ & \mathbf{D} \mathbf{x} \geq \mathbf{d}. \end{aligned} \tag{1.3}$$

The vector of decision variables \mathbf{x} in model (1.3) represents the portfolio holdings. These holdings typically represent the percentages invested in each asset and thus are often subject to the full investment constraint $\mathbf{1}^T \mathbf{x} = 1$. Other common constraints include the long-only constraint $\mathbf{x} \geq \mathbf{0}$, as well as restrictions related to sector or industry composition, turnover, etc. The terms $\mathbf{x}^T \mathbf{V} \mathbf{x}$ and $\boldsymbol{\mu}^T \mathbf{x}$ in the objective function are respectively the variance, which is a measure of risk,

¹ “Toward a theory of market value of risky assets”. Unpublished manuscript, 1961.

and the expected return of the portfolio defined by \mathbf{x} . The risk-aversion constant $\gamma > 0$ in the objective determines the tradeoff between risk and return of the portfolio.

Risk Management

Risk is inherent in most economic activities. This is especially true of financial activities where results of decisions made today may have many possible different outcomes depending on future events. Since companies cannot usually insure themselves completely against risk, they have to manage it. This is a hard task even with the support of advanced mathematical techniques. Poor risk management led to several spectacular failures in the financial industry in the 1990s (e.g., Barings Bank, Long Term Capital Management, Orange County). It was also responsible for failures and bailouts of a number of institutions (e.g., Lehman Brothers, Bear Stearns, AIG) during the far more severe global financial crisis of 2007–2008. Regulations, such as those prescribed by the Basel Accord (see Basel Committee on Banking Supervision, 2011), mandate that financial institutions control their risk via a variety of measurable requirements. The modeling of regulatory constraints as well as other risk-related constraints that the firm wishes to impose to prevent vulnerabilities can often be stated as a set of constraints

$$\mathbf{RM}(\mathbf{x}) \leq \mathbf{b}. \quad (1.4)$$

The vector \mathbf{x} in (1.4) represents the holdings in a set of risky securities. The entries of the vector-valued function $\mathbf{RM}(\mathbf{x})$ represent one or more measures of risk and the vector \mathbf{b} represents the acceptable upper limits on these measures. The set of risk management constraints (1.4) may be embedded in a more elaborate model that aims to optimize some kind of performance measure such as expected investment return.

In Chapter 2 we discuss a linear programming model for optimal bank planning under Basel III regulations. In this case the components of the function $\mathbf{RM}(\mathbf{x})$ are linear functions of \mathbf{x} . In Chapter 11 we discuss more sophisticated risk measures such as value at risk and conditional value at risk that typically make $\mathbf{RM}(\mathbf{x})$ a nonlinear function of \mathbf{x} .

Asset and Liability Management

How should a financial institution manage its assets and liabilities? A static model, such as the Markowitz mean–variance portfolio selection model, fails to incorporate the multi-period nature of typical liabilities faced by financial institutions. Furthermore, it penalizes returns both above and below the mean. A multi-period model that emphasizes the need to meet liabilities in each period for a finite (or possibly infinite) horizon is often more appropriate. Since liabilities and asset returns usually have random components, their optimal management requires techniques to optimize under uncertainty such as stochastic optimization.

We discuss several asset and liability management models in [Chapters 3, 16, and 17](#). A generic asset and liability management model can often be formulated as a stochastic programming problem of the form

$$\begin{aligned} \max_{\mathbf{x}} \quad & \mathbb{E}(U(\mathbf{x})) \\ & \mathbf{F}\mathbf{x} = \mathbf{L} \\ & \mathbf{D}\mathbf{x} \geq \mathbf{0}. \end{aligned} \tag{1.5}$$

The vector \mathbf{x} in (1.5) represents the investment decisions for the available assets at the dates in the planning horizon. The vector \mathbf{L} in (1.5) represents the liabilities that the institution faces at the dates in the planning horizon. The constraints $\mathbf{F}\mathbf{x} = \mathbf{L}$, $\mathbf{D}\mathbf{x} \geq \mathbf{0}$ represent the cash flow rules and restrictions applicable to the assets during the planning horizon. The term $U(\mathbf{x})$ in the objective function is some appropriate measure of utility. For instance, it could be the value of terminal wealth at the end of the planning horizon. In general, the components $\mathbf{F}, \mathbf{L}, \mathbf{D}$ are discrete-time random processes and thus (1.5) is a multi-stage stochastic programming model with recourse. In [Chapter 3](#) we discuss some special cases of (1.5) with no randomness.

1.4 Notes

George Dantzig was the inventor of linear programming and author of many related articles as well as a classical reference on the subject (Dantzig, 1963). A particularly colorful and entertaining description of the diet problem, a classical linear programming model, can be found in Dantzig (1990).

Boyd and Vandenberghe (2004) give an excellent exposition of convex optimization appropriate for senior or first-year graduate students in engineering. This book is freely available at:

www.stanford.edu/~boyd/cvxbook/

Ragsdale (2007) gives a practical exposition of optimization and related spreadsheet models that circumvent most technical issues. It is appropriate for senior or Master's students in business.

2 Linear Programming: Theory and Algorithms

Linear programming is one of the most significant contributions to computational mathematics made in the twentieth century. This chapter introduces the main ideas behind linear programming theory and algorithms. It also introduces two easy-to-use solvers.

2.1 Linear Programming

A *linear program* is an optimization problem whose objective is to minimize or maximize a linear function subject to a finite set of linear equality and linear inequality constraints. By flipping signs if necessary, a linear program can always be written in the generic form:

$$\begin{array}{ll}\min_{\mathbf{x}} & \mathbf{c}^T \mathbf{x} \\ \text{s.t.} & \mathbf{A} \mathbf{x} = \mathbf{b} \\ & \mathbf{D} \mathbf{x} \geq \mathbf{d}\end{array}$$

for some vectors and matrices $\mathbf{c} \in \mathbb{R}^n$, $\mathbf{b} \in \mathbb{R}^m$, $\mathbf{d} \in \mathbb{R}^p$, $\mathbf{A} \in \mathbb{R}^{m \times n}$, $\mathbf{D} \in \mathbb{R}^{p \times n}$. The terms *linear programming model* or *linear optimization model* are also used to refer to a linear program. We will use these terms interchangeably throughout the book.

The following two simplified portfolio construction examples illustrate the use of linear programming as a modeling tool.

Example 2.1 (Fund allocation) You would like to allocate \$80,000 among four mutual funds that have different expected returns as well as different weights in large-, medium- and small-capitalization stocks.

Capitalization	Fund 1	Fund 2	Fund 3	Fund 4
Large	50%	30%	25%	60%
Medium	30%	10%	40%	20%
Small	20%	60%	35%	20%
Exp. return	10%	15%	16%	8%

The allocation must contain at least 35% large-cap, 30% mid-cap, and 15% small-cap stocks. Find an acceptable allocation with the highest expected return assuming you are only allowed to hold long positions in the funds.

This problem can be formulated as the following linear programming model.

Linear programming model for fund allocation

Variables:

x_i : amount (in \$1000s) invested in fund i for $i = 1, \dots, 4$.

Objective:

$$\max \quad 0.10x_1 + 0.15x_2 + 0.16x_3 + 0.08x_4.$$

Constraints:

$$\begin{aligned} 0.50x_1 + 0.30x_2 + 0.25x_3 + 0.60x_4 &\geq 0.35 * 80 && \text{(large-cap)} \\ 0.30x_1 + 0.10x_2 + 0.40x_3 + 0.20x_4 &\geq 0.30 * 80 && \text{(mid-cap)} \\ 0.20x_1 + 0.60x_2 + 0.35x_3 + 0.20x_4 &\geq 0.15 * 80 && \text{(small-cap)} \\ x_1 + x_2 + x_3 + x_4 &= 80 && \text{(money to allocate)} \\ x_1, \dots, x_4 &\geq 0 && \text{(long-only positions).} \end{aligned}$$

Example 2.2 (Bond allocation) A bond portfolio manager has \$100,000 to allocate to two different bonds: a corporate bond and a government bond. These bonds have the following yield, risk level, and maturity:

Bond	Yield	Risk level	Maturity
Corporate	4%	2	3 years
Government	3%	1	4 years

The portfolio manager would like to allocate the funds so that the average risk level of the portfolio is at most 1.5 and the average maturity is at most 3.6 years. Any amount not invested in the bonds will be kept in a cash account that is assumed to generate no interest and does not contribute to the average risk level or maturity. In other words, assume cash has zero yield, zero risk level, and zero maturity.

How should the manager allocate funds to the two bonds to maximize yield? Assume the portfolio can only include long positions.

This problem can be formulated as the following linear programming model.

Linear programming model for bond allocation

Variables:

x_1, x_2 : amounts (in \$1000s) invested in the corporate and government bonds respectively.

Objective:

$$\max \quad 4x_1 + 3x_2.$$

Constraints:

$$\begin{aligned}
 x_1 + x_2 &\leq 100 && \text{(total funds)} \\
 \frac{2x_1 + x_2}{100} &\leq 1.5 && \text{(risk level)} \\
 \frac{3x_1 + 4x_2}{100} &\leq 3.6 && \text{(maturity)} \\
 x_1, x_2 &\geq 0 && \text{(long-only positions)}
 \end{aligned}$$

or equivalently

$$\begin{aligned}
 \max \quad & 4x_1 + 3x_2 \\
 \text{s.t.} \quad & \\
 & x_1 + x_2 \leq 100 \quad \text{(total funds)} \\
 & 2x_1 + x_2 \leq 150 \quad \text{(risk level)} \\
 & 3x_1 + 4x_2 \leq 360 \quad \text{(maturity)} \\
 & x_1, x_2 \geq 0 \quad \text{(long-only positions)}.
 \end{aligned}$$

The linear programming model in [Example 2.1](#) can be written more concisely using matrix–vector notation as follows:

$$\begin{aligned}
 \max \quad & \mathbf{r}^\top \mathbf{x} \\
 \text{s.t.} \quad & \mathbf{A}\mathbf{x} = \mathbf{b} \\
 & \mathbf{D}\mathbf{x} \geq \mathbf{d} \\
 & \mathbf{x} \geq \mathbf{0},
 \end{aligned}$$

$$\text{where } \mathbf{r} = \begin{bmatrix} 0.10 \\ 0.15 \\ 0.16 \\ 0.08 \end{bmatrix}, \mathbf{A} = \begin{bmatrix} 1 & 1 & 1 & 1 \end{bmatrix}, \mathbf{b} = 80, \mathbf{D} = \begin{bmatrix} 0.5 & 0.3 & 0.25 & 0.6 \\ 0.3 & 0.1 & 0.4 & 0.2 \\ 0.2 & 0.6 & 0.35 & 0.2 \end{bmatrix}, \text{ and } \mathbf{d} = \begin{bmatrix} 28 \\ 24 \\ 12 \end{bmatrix}.$$

Likewise, the linear programming model in [Example 2.2](#) can be written as

$$\begin{aligned}
 \max \quad & \mathbf{r}^\top \mathbf{x} \\
 \text{s.t.} \quad & \mathbf{A}\mathbf{x} \leq \mathbf{b} \\
 & \mathbf{x} \geq \mathbf{0},
 \end{aligned}$$

$$\text{for } \mathbf{r} = \begin{bmatrix} 4 \\ 3 \end{bmatrix}, \mathbf{A} = \begin{bmatrix} 1 & 1 \\ 2 & 1 \\ 3 & 4 \end{bmatrix}, \text{ and } \mathbf{b} = \begin{bmatrix} 100 \\ 150 \\ 360 \end{bmatrix}.$$

A linear programming model is in *standard form* if it is written as follows:

$$\begin{aligned}
 \min \quad & \mathbf{c}^\top \mathbf{x} \\
 \text{s.t.} \quad & \mathbf{A}\mathbf{x} = \mathbf{b} \\
 & \mathbf{x} \geq \mathbf{0}.
 \end{aligned}$$

The standard form is a kind of formatting convention that is used by some solvers. It is also particularly convenient to describe the most popular algorithms for solving linear programming, namely the simplex and interior-point methods.

The standard form is not restrictive. Any linear program can be rewritten in standard form. In particular, inequality constraints (other than non-negativity) can be rewritten as equality constraints after the introduction of a so-called *slack* or *surplus* variable. For instance, the linear program from [Example 2.2](#) can be written as

$$\begin{array}{llllll}
 \max & 4x_1 + 3x_2 & & & & \\
 \text{s.t.} & & & & & \\
 & x_1 + x_2 + x_3 & & & & = 100 \\
 & 2x_1 + x_2 & & + x_4 & & = 150 \\
 & 3x_1 + 4x_2 & & & + x_5 & = 360 \\
 & x_1, x_2, x_3, x_4, x_5 & & & & \geq 0.
 \end{array}$$

More generally, a linear program of the form

$$\begin{array}{ll}
 \min & \mathbf{c}^\top \mathbf{x} \\
 \text{s.t.} & \mathbf{Ax} \leq \mathbf{b} \\
 & \mathbf{x} \geq \mathbf{0}
 \end{array}$$

can be rewritten as

$$\begin{array}{ll}
 \min & \mathbf{c}^\top \mathbf{x} \\
 \text{s.t.} & \mathbf{Ax} + \mathbf{s} = \mathbf{b} \\
 & \mathbf{x}, \mathbf{s} \geq \mathbf{0}.
 \end{array}$$

It can then be rewritten, using matrix notation, in the following standard form:

$$\begin{array}{ll}
 \min & \begin{bmatrix} \mathbf{c} \\ \mathbf{0} \end{bmatrix}^\top \begin{bmatrix} \mathbf{x} \\ \mathbf{s} \end{bmatrix} \\
 \text{s.t.} & \begin{bmatrix} \mathbf{A} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \mathbf{s} \end{bmatrix} = \mathbf{b} \\
 & \begin{bmatrix} \mathbf{x} \\ \mathbf{s} \end{bmatrix} \geq \mathbf{0}.
 \end{array}$$

Unrestricted variables can be expressed as the difference of two new non-negative variables. For example, consider the linear program

$$\begin{array}{ll}
 \min & \mathbf{c}^\top \mathbf{x} \\
 \text{s.t.} & \mathbf{Ax} \leq \mathbf{b}.
 \end{array}$$

The unrestricted variable \mathbf{x} can be replaced by $\mathbf{u} - \mathbf{v}$ where $\mathbf{u}, \mathbf{v} \geq \mathbf{0}$. Hence the above linear program can be rewritten as

$$\begin{array}{ll}
 \min & \mathbf{c}^\top (\mathbf{u} - \mathbf{v}) \\
 \text{s.t.} & \mathbf{A}(\mathbf{u} - \mathbf{v}) \leq \mathbf{b} \\
 & \mathbf{u}, \mathbf{v} \geq \mathbf{0}.
 \end{array}$$

It can also be rewritten, after adding slack variables and using matrix notation, in the following standard form:

$$\begin{aligned} \min \quad & \begin{bmatrix} \mathbf{c} \\ -\mathbf{c} \\ \mathbf{0} \end{bmatrix}^T \begin{bmatrix} \mathbf{u} \\ \mathbf{v} \\ \mathbf{s} \end{bmatrix} \\ \text{s.t.} \quad & \begin{bmatrix} \mathbf{A} & -\mathbf{A} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \mathbf{v} \\ \mathbf{s} \end{bmatrix} = \mathbf{b} \\ & \begin{bmatrix} \mathbf{u} \\ \mathbf{v} \\ \mathbf{s} \end{bmatrix} \geq \mathbf{0}. \end{aligned}$$

2.2 Graphical Interpretation of a Two-Variable Example

Banks need to consider regulations when determining their business strategy. In this section, we consider the Basel III regulations (Basel Committee on Banking Supervision, 2011). We present a simplified example following the paper of Pokutta and Schmaltz (2012). Consider a bank with total deposits D and loans L . The loans may default and the deposits are exposed to early withdrawal. The bank holds capital C in order to buffer against possible default losses on the loans, and it holds a liquidity reserve R to buffer against early withdrawals on the deposits. The balance sheet of the bank satisfies $L + R = D + C$. Normalizing the total assets to 1, we have $R = 1 - L$ and $C = 1 - D$. Basel III regulations require banks to satisfy four minimum ratio constraints in order to buffer against different types of risk:

Capital ratio: $\frac{C}{L} \geq r_1$

Leverage ratio: $C \geq r_2$

Liquidity coverage ratio: $\frac{R}{D} \geq r_3$

Net stable funding ratio: $\frac{\alpha D + C}{L} \geq r_4,$

where the ratios $r_1, r_2, r_3, r_4, \alpha$ are computed for each bank based on the riskiness of its loans and the likelihood of early withdrawals on deposits. To illustrate, consider a bank with $r_1 = 0.3$, $r_2 = 0.1$, $r_3 = 0.25$, $r_4 = 0.7$, $\alpha = 0.3$. Expressing the four ratio constraints in terms of the variables D and L , we get

$$\begin{aligned} D + 0.3L &\leq 1 \\ D &\leq 0.9 \\ 0.25D + L &\leq 1 \\ 0.7D + 0.7L &\leq 1. \end{aligned}$$

Figure 2.1 displays a plot of the feasible region of this system of inequalities in the plane (D, L) .

Given this feasible region, the objective of the bank is to maximize the margin income $m_D D + m_L L$ that it makes on its products; where m_D is the margin that

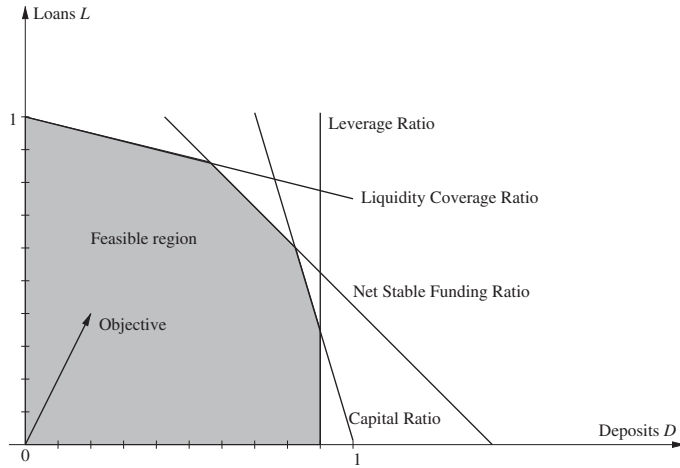


Figure 2.1 Basel III regulations

the bank makes on its deposits and m_L is the margin charged on its loans. For example, if $m_D = 0.02$ and $m_L = 0.03$, the best solution that satisfies all the constraints corresponds to the vertex $D = 0.571, L = 0.857$ on the boundary of the feasible region, at the intersection of the lines $0.25D + L = 1$ and $0.7D + 0.7L = 1$. This means that the bank should have 57.1% of its liabilities in deposits and 42.9% in capital, and it should have 85.7% of its assets in loans and the remaining 14.3% in liquidity reserve. The fact that an optimal solution occurs at a vertex of the feasible region is a property of linear programs that extends to higher dimensions than 2: To find an optimal solution of a linear program, it suffices to restrict the search to vertices of the feasible region. This geometric insight is the basis of the simplex method, which goes from one vertex of the feasible region to an adjacent one with a better objective value until it reaches an optimum. An algebraic description of the simplex method that can be coded on a computer is presented in [Section 2.7.1](#).

2.3 Numerical Linear Programming Solvers

There are a variety of both commercial and open-source software packages for linear programming. Most of these packages implement the algorithms described in [Section 2.7](#) below. Next we illustrate two of these solvers by applying them to [Example 2.1](#).

Excel Solver

[Figure 2.2](#) displays a printout of an Excel spreadsheet implementation of the linear programming model for [Example 2.1](#) as well as the dialog box obtained when we run the Excel add-in **Solver**. The spreadsheet model contains the three

components of the linear program. The decision variables are in the range B4:E4. The objective is in cell F3. The left- and right-hand sides of the equality constraint are in the cells F4 and H4 respectively. Likewise, the left- and right-hand sides of the three inequality constraints are in the ranges F8:F10 and H8:H10 respectively. These components are specified in the **Solver** dialog box. In addition, the **Solver** options are used to indicate that this is a linear model and that the variables are non-negative.

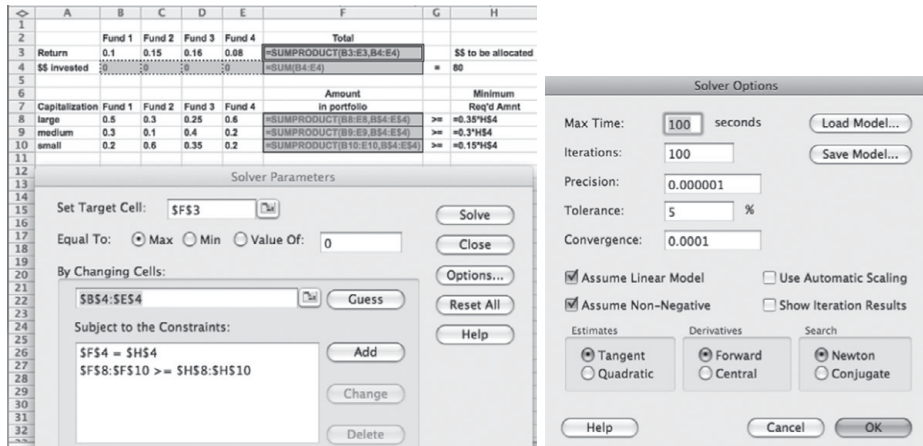


Figure 2.2 Spreadsheet implementation and the Solver dialog box for the fund allocation model

MATLAB CVX

Figure 2.3 displays a CVX script for the same problem. The script can be run provided the freely available CVX toolbox is installed.

Either Excel **Solver** or MATLAB CVX find the following optimal solution to the problem in Example 2.1:

$$\mathbf{x}^* = \begin{bmatrix} 0.0000 \\ 12.6316 \\ 46.3158 \\ 21.0526 \end{bmatrix},$$

and the corresponding optimal objective value 10.9895 (recall that the units are in \$1000s).

2.4 Sensitivity Analysis

In addition to the optimal solution, the process of solving a linear program also generates some interesting *sensitivity information* via the so-called *shadow prices*

```
File Edit Text Go Cell Tools Debug Desktop Window
x [ ] - 1.0 + + 1.1 x [ ] [ ] [ ]
1 % Matlab CVX code for the fund allocation problem
2 n = 4 ;
3 r = [0.10;0.15;0.16;0.08] ;
4 A = [1 1 1 1] ;
5 b = 80 ;
6 D = [0.5 0.3 0.25 0.6;
7       0.3 0.1 0.4 0.2;
8       0.2 0.6 0.35 0.2] ;
9 d = [28;24;12] ;
10
11 cvx_begin
12     variables x(n) ;
13     maximize (r'*x)
14     A*x == b ;
15     D*x >= d ;
16     x >= zeros(n,1) ;
17 cvx_end
18
```

Figure 2.3 MATLAB CVX code for the fund allocation model

or *dual values* associated with the constraints. Assume that the constraints of a linear program, and hence the shadow prices, are indexed by $i = 1, \dots, m$. The *shadow price* y_i^* of the i th constraint has the following sensitivity interpretation:

If the right-hand side of the i th constraint changes by Δ , then the optimal value of the linear program changes by $\Delta \cdot y_i^*$ as long as Δ is within a certain range.

Both Excel Solver and MATLAB CVX compute the shadow prices implicitly. To make this information explicit in Excel Solver we request a sensitivity report after running it as shown in Figure 2.4.

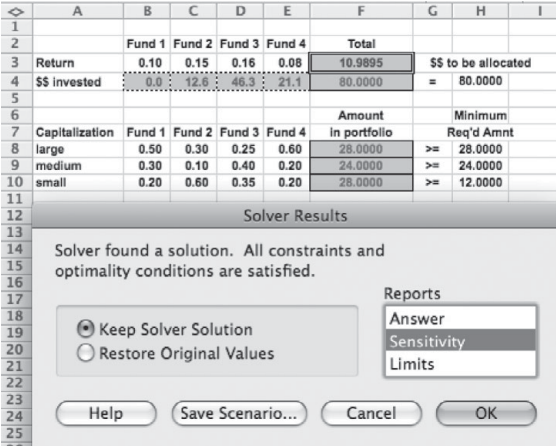


Figure 2.4 Requesting sensitivity report in Solver

Figure 2.5 displays the sensitivity report for Example 2.1.

Adjustable Cells

Cell	Name	Final Value	Reduced Cost	Objective Coefficient	Allowable Increase	Allowable Decrease
\$B\$4	\$\$ invested Fund 1	0	-0.00263158	0.1	0.00263158	1.00E+30
\$C\$4	\$\$ invested Fund 2	12.63157895	0	0.15	0.0166667	0.00142857
\$D\$4	\$\$ invested Fund 3	46.31578947	0	0.16	0.00166667	0.00625
\$E\$4	\$\$ invested Fund 4	21.05263158	0	0.08	0.01	0.00357143

Constraints

Cell	Name	Final Value	Shadow Price	Constraint R.H. Side	Allowable Increase	Allowable Decrease
\$F\$4	\$\$ invested Total	80.0000	0.22	80	21.0526	6.31579
\$F\$8	large in portfolio	28.0000	-0.231579	28	6	6.66667
\$F\$9	medium in portfolio	24.0000	-0.00526316	24	3.42857	14.6667
\$F\$10	small in portfolio	28.0000	0	12	16	1.00E+30

Figure 2.5 Sensitivity report

The values y_i^* can be found in the column labeled “Shadow Price”. In addition, the “Allowable Increase” and “Allowable Decrease” columns indicate the range of change for each right-hand side of a constraint where the sensitivity analysis holds. For example, if the right-hand side of the large-capitalization constraint

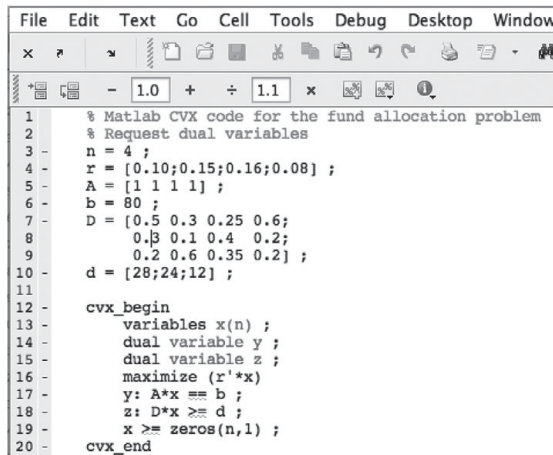
$$0.5\mathbf{x}_1 + 0.3\mathbf{x}_2 + 0.25\mathbf{x}_3 + 0.6\mathbf{x}_4 \geq 28$$

changes from 28 to $28 + \Delta$, then the optimal value changes by $-0.231579 \cdot \Delta$. This holds provided Δ is within the allowable range $[-6.6666, 6]$. If the requirement on large-cap stocks is reduced from 35% to 30%, the change in right-hand side is $\Delta = -0.05 \cdot 80 = -4$, which is within the allowable range. Therefore the optimal objective value increases by $-0.231579 \cdot (-4) = 0.926316$. Because our units are in \$1000, this means that the expected return on an optimal portfolio would increase by \$926.32 if we relaxed the constraint on large-cap stocks by 5%, from 35% to 30%.

The shadow prices of the non-negativity constraints are the “Reduced Cost” displayed in the initial part of the sensitivity report. This is also the convention for more general lower and upper bounds on the decision variables. Observe that in [Example 2.1](#) the reduced costs of the non-zero variables are zero. The reduced costs also have a deeper meaning in the context of the simplex algorithm for linear programming as described in [Section 2.7.1](#) below.

A linear programming model is *non-degenerate* if all of the allowable increase and allowable decrease limits are positive. The above linear programming model is non-degenerate.

In **CVX** this information can also be obtained by including a few additional pieces of code to save the dual information in the dual variables \mathbf{y}, \mathbf{z} as shown in [Figure 2.6](#).



```

1 % Matlab CVX code for the fund allocation problem
2 % Request dual variables
3 n = 4 ;
4 r = [0.10;0.15;0.16;0.08] ;
5 A = [1 1 1 1] ;
6 b = 80 ;
7 D = [0.5 0.3 0.25 0.6;
8      0.3 0.1 0.4 0.2;
9      0.2 0.6 0.35 0.2] ;
10 d = [28;24;12] ;
11
12 cvx_begin
13     variables x(n) ;
14     dual variable y ;
15     dual variable z ;
16     maximize (r'*x)
17     y: A*x == b ;
18     z: D*x >= d ;
19     x >= zeros(n,1) ;
20 cvx_end

```

Figure 2.6 MATLAB CVX code with dual variables

Both solvers yield the following dual values: $\mathbf{y}^* = 0.22$, $\mathbf{z}^* = \begin{bmatrix} -0.231579 \\ -0.005263 \\ 0 \end{bmatrix}$.

We note that some solvers may flip the sign of the dual values. In particular, the output of the above CVX code yields the values -0.22 and $\begin{bmatrix} 0.231579 \\ 0.005263 \\ 0 \end{bmatrix}$. It is

important to be mindful of this subtlety when interpreting the dual information. The ambiguity can be easily resolved by thinking in terms of sensitivity analysis. In this particular example, it is clear that the shadow price of the first constraint should be non-negative as more capital should lead to a higher return. Likewise, it is clear that the shadow prices of the other constraints should be non-positive as more stringent diversification constraints, e.g., higher percentage in large cap, reduces the set of feasible portfolios and hence can only lead to portfolios with return less than or equal to the optimal return of the original problem.

2.5 *Duality

Every linear program has an associated *dual* linear programming problem. The properties of these two linear programs and how they are related to each other have deep implications. In particular, duality enables us to answer the following kinds of questions:

- Can we recognize an optimal solution?
- Can we construct an algorithm to find an optimal solution?
- Can we assess how suboptimal a current feasible solution is?

The attentive reader may have noticed that dual variables were already mentioned in [Section 2.4](#) when discussing sensitivity analysis with CVX. This is not a

coincidence. There is a close connection between duality and sensitivity analysis. The vector of shadow prices of the constraints of a linear program corresponds precisely to the optimal solution of its dual.

Consider the following linear program in standard form, which we shall refer to as the *primal* problem:

$$\begin{aligned} \min \quad & \mathbf{c}^\top \mathbf{x} \\ \text{s.t.} \quad & \mathbf{Ax} = \mathbf{b} \\ & \mathbf{x} \geq \mathbf{0}. \end{aligned} \quad (2.1)$$

The following linear program is called the *dual* problem:

$$\begin{aligned} \max \quad & \mathbf{b}^\top \mathbf{y} \\ \text{s.t.} \quad & \mathbf{A}^\top \mathbf{y} \leq \mathbf{c}. \end{aligned} \quad (2.2)$$

Sometimes it is convenient to rewrite the constraints in the dual problem as equality constraints by means of slack variables. That is, problem (2.2) can also be written as

$$\begin{aligned} \max \quad & \mathbf{b}^\top \mathbf{y} \\ \text{s.t.} \quad & \mathbf{A}^\top \mathbf{y} + \mathbf{s} = \mathbf{c} \\ & \mathbf{s} \geq \mathbf{0}. \end{aligned} \quad (2.3)$$

There is a deep connection between the primal and dual problems. The next result follows by construction.

Theorem 2.3 (Weak duality) *Assume \mathbf{x} is a feasible point for (2.1) and \mathbf{y} is a feasible point for (2.2). Then*

$$\mathbf{b}^\top \mathbf{y} \leq \mathbf{c}^\top \mathbf{x}.$$

Proof Under the assumptions on \mathbf{x} and \mathbf{y} it follows that

$$\mathbf{b}^\top \mathbf{y} = (\mathbf{Ax})^\top \mathbf{y} = (\mathbf{A}^\top \mathbf{y})^\top \mathbf{x} \leq \mathbf{c}^\top \mathbf{x}. \quad \square$$

The following (not so straightforward) result also holds.

Theorem 2.4 (Strong duality) *Assume one of the problems (2.1) or (2.2) is feasible. Then this problem is bounded if and only if the other one is feasible. In that case both problems have optimal solutions and their optimal values are the same.*

We refer the reader to Bertsimas and Tsitsiklis (1997) or Chvátal (1983) for a proof of Theorem 2.4. This result is closely related to the following classical properties of linear inequality systems.

Theorem 2.5 *Assume $\mathbf{A} \in \mathbb{R}^{m \times n}$ and $\mathbf{b} \in \mathbb{R}^m$. In each of the following cases exactly one of the systems (I) or (II) has a solution but not both.*

(a) *Farkas's lemma*

$$\mathbf{Ax} = \mathbf{b}, \mathbf{x} \geq \mathbf{0}, \quad (\text{I})$$

$$\mathbf{A}^\top \mathbf{y} \leq \mathbf{0}, \mathbf{b}^\top \mathbf{y} < 0. \quad (\text{II})$$

(b) *Gordan's theorem*

$$\mathbf{Ax} = \mathbf{0}, \mathbf{x} \not\geq \mathbf{0}, \quad (\text{I})$$

$$\mathbf{A}^\top \mathbf{y} > \mathbf{0}. \quad (\text{II})$$

(c) *Stiemke's theorem*

$$\mathbf{Ax} = \mathbf{0}, \mathbf{x} > \mathbf{0}, \quad (\text{I})$$

$$\mathbf{A}^\top \mathbf{y} \not\geq \mathbf{0}. \quad (\text{II})$$

The equivalence between [Theorems 2.4](#) and [2.5](#) is explored in [Exercises 2.11](#) and [2.12](#).

We next present a derivation of the dual problem via the so-called Lagrangian function. This derivation has the advantage of introducing an important concept that we will encounter again in later chapters. Associated with the optimization problem (2.1) consider the *Lagrangian* function defined by

$$L(\mathbf{x}, \mathbf{y}, \mathbf{s}) := \mathbf{c}^\top \mathbf{x} + \mathbf{y}^\top (\mathbf{b} - \mathbf{Ax}) - \mathbf{s}^\top \mathbf{x}.$$

The constraints of (2.1) can be encoded using the Lagrangian function via the following observation: For a given vector \mathbf{x}

$$\max_{\substack{\mathbf{y}, \mathbf{s} \\ \mathbf{s} \geq \mathbf{0}}} L(\mathbf{x}, \mathbf{y}, \mathbf{s}) = \begin{cases} \mathbf{c}^\top \mathbf{x} & \text{if } \mathbf{Ax} = \mathbf{b} \text{ and } \mathbf{x} \geq \mathbf{0} \\ +\infty & \text{otherwise.} \end{cases}$$

Therefore the primal problem (2.1) can be written as

$$\min_{\mathbf{x}} \max_{\substack{\mathbf{y}, \mathbf{s} \\ \mathbf{s} \geq \mathbf{0}}} L(\mathbf{x}, \mathbf{y}, \mathbf{s}). \quad (2.4)$$

On the other hand, observe that $L(\mathbf{x}, \mathbf{y}, \mathbf{s}) = \mathbf{b}^\top \mathbf{y} + (\mathbf{c} - \mathbf{A}^\top \mathbf{y} - \mathbf{s})^\top \mathbf{x}$. Hence for a given pair of vectors (\mathbf{y}, \mathbf{s})

$$\min_{\mathbf{x}} L(\mathbf{x}, \mathbf{y}, \mathbf{s}) = \begin{cases} \mathbf{b}^\top \mathbf{y} & \text{if } \mathbf{A}^\top \mathbf{y} + \mathbf{s} = \mathbf{c} \\ -\infty & \text{otherwise.} \end{cases}$$

The dual problem is obtained by flipping the order of the min and max operations in (2.4). Indeed, observe that the dual problem (2.3) can be written as

$$\max_{\substack{\mathbf{y}, \mathbf{s} \\ \mathbf{s} \geq \mathbf{0}}} \min_{\mathbf{x}} L(\mathbf{x}, \mathbf{y}, \mathbf{s}).$$

A similar procedure can be applied to obtain the dual of a linear program that is not necessarily in standard form. For example, the primal problem

$$\begin{aligned} \min \quad & \mathbf{c}^\top \mathbf{x} \\ \text{s.t.} \quad & \mathbf{Ax} \geq \mathbf{b} \\ & \mathbf{x} \geq \mathbf{0} \end{aligned} \quad (2.5)$$

can be written as

$$\min_{\mathbf{x}} \max_{\mathbf{y} \geq \mathbf{0}, \mathbf{s} \geq \mathbf{0}} L(\mathbf{x}, \mathbf{y}, \mathbf{s}),$$