Advanced Lecture on Mathematical Science and Information Science I

Optimization in Finance

Reha H. Tütüncü
Visiting Associate Professor
Dept. of Mathematical and Computing Sciences
Tokyo Institute of Technology

Department of Mathematical Sciences
Carnegie Mellon University
Pittsburgh, PA 15213 USA
e-mail: reha@cmu.edu

Spring 2003

Contents

List of Figures

List of Tables

Preface

Optimization models and methods play an increasingly important role in financial decisions. Many computational finance problems ranging from asset allocation to risk management, from option pricing to model calibration can be solved efficiently using modern optimization techniques. This manuscript discusses several classes of optimization problems (including linear, quadratic, conic, robust, and stochastic programming problems) encountered in financial models. 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. In addition to classical and well-known models such as Markowitz' mean-variance optimization formulation we present some newer optimization models for a variety of financial problems.

This manuscript is derived from a set of course notes I prepared for the course *Advanced Lecture on Mathematical Science and Information Science I: Optimization in Finance* that I taught at the Department of Mathematical and Computing Sciences at Tokyo Institute of Technology between April 18, 2003 and July 18, 2003, during my sabbatical visit to Tokyo Tech. Parts of these notes are based on the lectures I presented at the University of Coimbra, Portugal in the Summer of 2002 as part of a short course I taught there. I gratefully acknowledge the financial support of these two institutions during my stays. I also thank the attendants of these courses and, in particular, my hosts Luís N. Vicente in Coimbra and Masakazu Kojima in Tokyo, for their feedback and for many stimulating discussions.

Reha H. Tütüncü
August 2003, Tokyo

Chapter 1

Introduction

Optimization is a branch of applied mathematics that derives its importance both from the wide variety of its applications and from the availability of advanced algorithms for the efficient and robust solution of many of its problem classes. Mathematically, it refers to the minimization (or maximization) of a given *objective function* of several *decision variables* that have to satisfy some functional *constraints*. A typical optimization model addresses the allocation of scarce resources among a set of alternative activities in order to maximize an objective function–a measure of the modeler's satisfaction with the solution, for example, the total profit.

Decision variables, the objective function, and constraints are three essential elements of any optimization problem. Some problems may lack constraints so that any set of decision variables (of appropriate dimension) are acceptable as alternative solutions. Such problems are called *unconstrained optimization* problems, while others are often referred to as *constrained optimization* problems. There are problem instances with no objective functions–the so-called *feasibility* problems, and others with multiple objective functions. Such problems are often addressed by reduction to a single or a sequence of single-objective optimization problems.

If the decision variables in an optimization problem are restricted to integers, or to a discrete set of possibilities, we have an *integer* or *discrete optimization* problem. If there are no such restrictions on the variables, the problem is a *continuous optimization* problem. Of course, some problems may have a mixture of discrete and continuous variables. Our focus in these lectures will be on continuous optimization problems. We continue with a short classification of the problem classes we will encounter during our lectures.

1.1 Continuous Optimization: A Brief Classification

We start with a generic description of an optimization problem. Given a function $f(x) : \Re^n \to \Re$ and a set $S \subset \Re^n$, the problem of finding an $x^* \in \Re^n$ that solves

$$(\mathcal{OP}_0) \quad \begin{array}{ll} \min_x & f(x) \\ \text{s.t.} & x \in S \end{array} \tag{1.1}$$

is called an optimization problem (OP). We refer to f as the *objective function* and to S as the *feasible region*. If S is empty, the problem is called *infeasible*. If it is possible to find a sequence x^k such that $x^k \in S,\ \forall k$ and $f(x^k)$ diverges to $-\infty$, then the problem is *unbounded*. If the problem is neither infeasible nor unbounded, then it is often possible to find a solution x^* that satisfies

$$f(x^*) \leq f(x),\ \forall x \in S.$$

Such an x^* is called a *global minimizer* of the problem (OP). If

$$f(x^*) < f(x),\ \forall x \in S, x \neq x^*,$$

then x^* is a *strict global minimizer*. In other instances, we may only find an x^* that satisfies

$$f(x^*) \leq f(x),\ \forall x \in S \cap B_{x^*}(\varepsilon)$$

for some $\varepsilon > 0$ where $B_{x^*}(\varepsilon)$ is the open ball with radius ε centered at x^*, i.e.,

$$B_{x^*}(\varepsilon) = \{x : \|x - x^*\| < \varepsilon\}.$$

Such an x^* is called a *local minimizer* of the problem (OP). A *strict local minimizer* is defined similarly.

In most cases, the feasible set S is described explicitly using functional constraints (equalities and inequalities). For example, S may be given as

$$S := \{x : g_i(x) = 0, i \in \mathcal{E}, g_i(x) \geq 0, i \in \mathcal{I}\},$$

where $\mathcal{E}$ and $\mathcal{I}$ are the index sets for equality and inequality constraints. Then, our generic optimization problem takes the following form:

$$(\mathcal{OP}) \qquad \begin{array}{lll} \min_x & f(x) & \\ & g_i(x) = 0, & i \in \mathcal{E} \\ & g_i(x) \geq 0, & i \in \mathcal{I}. \end{array} \tag{1.2}$$

There are many factors that affect the efficient solvability of optimization problems. For example, n–the number of decision variables in the problem, and $|\mathcal{E}| + |\mathcal{I}|$–the total

number of constraints, are generally good predictors of how easy or difficult it would be to solve a given optimization problem. Other factors are related to the properties of the functions f and g_i's that define the problem. Problems with a linear objective function and linear constraints are easier, so are problems with convex objective functions and convex feasible sets. Therefore, instead of general purpose optimization algorithms, researchers have developed different algorithms for problems with special characteristics. This approach requires a proper classification of optimization problems. We list a few of these optimization problem classes that we will encounter in this manuscript. A more complete classification can be found, for example, on the *Optimization Tree* available from `http://www-fp.mcs.anl.gov/otc/Guide/OptWeb/`.

1.1.1 Linear Optimization

One of the most common and easiest optimization problems is the *linear optimization* (LO) or the *linear programming* (LP) problem: the problem of optimizing a linear objective function subject to linear equality and inequality constraints. This corresponds to the case where f and all g_i's are linear in $(\mathcal{OP})$. If either f or one of the g_i's is not linear, then the resulting problem is a *nonlinear programming* (NLP) problem.

The standard form of the LP is given below:

$$(\mathcal{LP}) \qquad \begin{array}{rcl} \min_x \quad c^T x & & \\ Ax & = & b \\ x & \geq & 0, \end{array} \tag{1.3}$$

where $A \in I\!R^{m\times n}$, $b \in I\!R^m$, $c \in I\!R^n$ are given, and $x \in I\!R^n$ is the variable vector to be determined as the solution to the problem.

As with $(\mathcal{OP})$, the problem $\mathcal{LP}$ is said to be *feasible* if its constraints are consistent and it is called *unbounded* if there exists a sequence of feasible vectors $\{x^k\}$ such that $c^T x^k \to -\infty$. When $\mathcal{LP}$ is feasible but not unbounded it has an *optimal solution*, i.e., a vector x that satisfies the constraints and minimizes the objective value among all feasible vectors.

The best known (and most successful) methods for solving LPs are *interior-point methods* and the *simplex method*.

1.1.2 Quadratic Optimization

A more general optimization problem is the *quadratic optimization* (QO) or the *quadratic programming* (QP) problem, where the objective function is now a quadratic function of the variables. The standard form QP is defined as follows:

$$(\mathcal{QP}) \qquad \begin{array}{rcl} \min_x \quad \frac{1}{2}x^T Q x + c^T x & & \\ Ax & = & b \\ x & \geq & 0, \end{array} \tag{1.4}$$

where $A \in I\!R^{m\times n}$, $b \in I\!R^m$, $c \in I\!R^n$, $Q \in I\!R^{n\times n}$ are given, and $x \in I\!R^n$. Since $x^TQx = \frac{1}{2}x^T(Q+Q^T)x$, Q can be assumed to be symmetric without loss of generality.

The objective function of the problem $\mathcal{QP}$ is a convex function of x when Q is a positive semidefinite matrix, i.e., when $y^TQy \geq 0$ for all y (see the Appendix for a discussion on convex functions). This condition is equivalent to Q having all nonnegative eigenvalues. When this condition is satisfied, the QP problem is a convex optimization problem and can be solved in polynomial time using interior-point methods.

1.1.3 Conic Optimization

Another generalization of the linear optimization problem is obtained by replacing the nonnegativity constraints with general conic inclusion constraints, resulting in a so-called *conic optimization* problem. For this purpose, we consider a closed convex cone C (see the Appendix for a brief discussion on cones) in a finite-dimensional vector space X and the following conic optimization problem:

$$(\mathcal{CO}) \qquad \begin{array}{rcl} \min_x \quad c^Tx & & \\ Ax & = & b \\ x \in C. & & \end{array} \tag{1.5}$$

When $X = \Re^n$ and $C = \Re^n_+$, this problem is the standard form LP. However, much more general nonlinear optimization problems can also be formulated in this way. Furthermore, some of the most efficient and robust algorithmic machinery developed for linear optimization problems can be modified to solve these general optimization problems. Two important subclasses of conic optimization problems we will address are: (i) second-order cone optimization, and (ii) semidefinite optimization. These correspond to the cases when C is the second-order cone:

$$C_q := \{x = (x_0, x_1, \ldots, x_n) \in \Re^{n+1} : x_0 \geq \|(x_1, \ldots, x_n)\|\},$$

and the cone of symmetric positive semidefinite matrices:

$$C_s := \left\{ X = \begin{bmatrix} x_{11} & \cdots & x_{1n} \\ \vdots & \ddots & \vdots \\ x_{n1} & \cdots & x_{nn} \end{bmatrix} \in \Re^{n\times n} : X = X^T,\ X \text{ is positive semidefinite} \right\}.$$

When we work with the cone of positive semidefinite matrices, the standard inner products used in c^Tx and Ax in (1.5) are replaced by an appropriate inner product for the space of n-dimensional square matrices.

1.2 Optimization with Data Uncertainty

In all the problem classes we discussed so far, we made the implicit assumption that the *data* of the problem, namely the parameters that describe the problem such as Q,

A, b and c in (QP) are all known. This is not always the case. Often, the problem parameters correspond to quantities that will only be realized in the future, or can not be known exactly at the time the problem has to be formulated and solved. Such situations are especially common in the models that involve financial quantities such as returns on investments, risks, etc. We will discuss two fundamentally different approaches that address optimization with data uncertainty. *Stochastic programming* is an approach used when the data uncertainty is random and can be explained by some probability distribution. *Robust optimization* is used when the uncertainty structure is not random and/or a solution that can behave well in all possible realizations of the uncertain data is desired. These two alternative approaches are not problem classes (as in LP, QP, etc.) but rather modeling techniques for addressing data uncertainty.

1.2.1 Stochastic Optimization

The term *stochastic optimization* or *stochastic programming* refers to an optimization problem in which some problem data are random. The underlying optimization problem might be a linear program, an integer program, or a nonlinear program. An important case is that of *stochastic linear programs.*

A stochastic program *with recourse* arises when some of the decisions (recourse actions) can be taken after the outcomes of some (or all) random events have become known. For example, a *two-stage stochastic linear program with recourse* can be written as follows:

$$\begin{array}{rrcrcl} \max & (c^1)^T x^1 & + & E[\max c^2(\omega)^T x^2(\omega)] & & \\ & A^1 x^1 & & & = & b^1 \\ & B^2(\omega) x^1 & + & A^2(\omega) x^2(\omega) & = & b^2(\omega) \\ & x^1 \geq 0, & & x^2(\omega) \geq 0, & & \end{array} \tag{1.6}$$

where the first-stage decisions are represented by vector x^1 and the second-stage decisions by vector $x^2(\omega)$, which depend on the realization ω of a random event. A^1 and b^1 define deterministic constraints on the first-stage decisions x^1, whereas $A^2(\omega)$, $B^2(\omega)$, and $b^2(\omega)$ define stochastic linear constraints linking the recourse decisions $x^2(\omega)$ to the first-stage decisions. The objective function contains a deterministic term $(c^1)^T x^1$ and the expectation of the second-stage objective $c^2(\omega)^T x^2(\omega)$ taken over all realization of the random event ω.

Note that, once the first-stage decisions x^1 have been made and the random event ω has been realized, one can compute the optimal second-stage decisions by solving the following linear program:

$$\begin{array}{rrcl} f(x^1, \omega) = \max & c^2(\omega)^T x^2(\omega) & & \\ & A^2(\omega) x^2(\omega) & = & b^2(\omega) - B^2(\omega) x^1 \\ & x^2(\omega) \geq 0, & & \end{array} \tag{1.7}$$

Let $f(x^1) = E[f(x^1, \omega)]$ denote the expected value of the optimal value of this problem.

Then, the two-stage stochastic linear program becomes

$$\begin{array}{rrcl} \max & (c^1)^T x^1 + f(x^1) & & \\ & A^1 x^1 & = & b^1 \\ & x^1 & \geq & 0, \end{array} \tag{1.8}$$

So, if the (possibly nonlinear) function $f(x^1)$ is known, the problem reduces to a nonlinear programming problem. When the data $c^2(\omega)$, $A^2(\omega)$, $B^2(\omega)$, and $b^2(\omega)$ are described by finite distributions, one can show that f is piecewise linear and concave. When the data are described by probability densities that are absolutely continuous and have finite second moments, one can show that f is differentiable and concave. In both cases, we have a convex optimization problem with linear constraints for which specialized algorithms are available.

1.2.2 Robust Optimization

Robust optimization refers to the modeling of optimization problems with data uncertainty to obtain a solution that is guaranteed to be "good" for all possible realizations of the uncertain parameters. In this sense, this approach departs from the randomness assumption used in stochastic optimization for uncertain parameters and gives the same importance to all possible realizations. Uncertainty in the parameters is described through *uncertainty sets* that contain all (or most) possible values that may be realized for the uncertain parameters.

There are different definitions and interpretations of robustness and the resulting models differ accordingly. One important concept is *model robustness*; this refers to solutions that remain *feasible* for all possible values of the uncertain inputs–we prefer to call such solutions *constraint robust solutions*. This type of solutions are required in many engineering applications. Here is an example adapted from Ben-Tal and Nemirovski: Consider a multi-phase engineering process (a chemical distillation process, for example) and a related process optimization problem that includes balance constraints (materials entering a phase of the process can not be more than what is produced/left over from the previous phase). Often, the quantities of the end products of a particular phase depend on external, uncontrollable factors and therefore are uncertain. However, no matter what the values of these uncontrollable factors are, the balance constraints *must* be satisfied. Therefore, our solution must be model robust with respect to the uncertainties of the problem. Here is a mathematical model for finding constraint robust solutions: Consider an optimization problem of the form:

$$(\mathcal{OP}_{uc}) \qquad \begin{array}{rrcl} \min_x & f(x) & & \\ & G(x,p) & \in & K. \end{array} \tag{1.9}$$

Here, x are the decision variables, f is the (certain) objective function, G and K are the structural elements of the constraints that are assumed to be certain and p are

the possibly uncertain parameters of the problem. Consider an uncertainty set $\mathcal{U}$ that contains all possible values of the uncertain parameters p. Then, a constraint robust optimal solution can be found by solving the following problem:

$$(\mathcal{CROP}) \qquad \begin{array}{lll} \min_x & f(x) & \\ & G(x,p) & \in \quad K, \forall p \in \mathcal{U}. \end{array} \tag{1.10}$$

Another important robustness concept is *solution robustness*. This refers to solutions that will remain close to optimal for all possible realizations of the uncertain problem parameters, and for this reason we prefer the alternative term *objective robust* for such solutions. Since such solutions may be difficult to obtain, especially when uncertainty sets are relatively large, an alternative goal for objective robustness is to find solutions whose worst-case behavior is optimized. Worst-case behavior of a solution corresponds to the value of the objective function for the worst possible realization of the uncertain data for that particular solution. Here is a mathematical model that addresses objective robustness: Consider an optimization problem of the form:

$$(\mathcal{OP}_{uo}) \qquad \begin{array}{lll} \min_x & f(x,p) & \\ & x & \in \quad S. \end{array} \tag{1.11}$$

Here, S is the (certain) feasible set and f is the objective function that depends on uncertain parameters p. Assume as above that $\mathcal{U}$ is the uncertainty set that contains all possible values of the uncertain parameters p. Then, an objective robust solution can be obtained by solving:

$$(\mathcal{ORCOP}) \qquad \min_{x \in S} \quad \max_{p \in \mathcal{U}} f(x,p). \tag{1.12}$$

Note that solution robustness is a special case of model robustness–it is easy to see that by introducing a new variable t (to be minimized) into ($\mathcal{OP}_{uo}$ and imposing the constraint $f(x,p) \leq t$, we get an equivalent problem to ($\mathcal{OP}_{uo}$; the constraint robust formulation of the resulting problem is equivalent to $\mathcal{ORCOP}$.

Model robustness and solution robustness are concepts that arise in conservative decision making and are not always appropriate for optimization problems with data uncertainty.

1.3 Financial Mathematics

Modern finance has become increasingly technical, requiring the use of sophisticated mathematical tools in both research and practice. Many find the roots of this trend in the portfolio selection models and methods described by Markowitz in the 1950's and the option pricing formulas developed by Black, Scholes, and Merton in the late 1960's. For the enormous effect these works produced on modern financial practice, Markowitz was awarded the Nobel prize in Economics in 1990, while Scholes and Merton won the Nobel prize in Economics in 1997.

Below, we list a number of topics in finance that are especially suited for mathematical analysis and involve sophisticated tools from mathematical sciences.

1.3.1 Portfolio Selection and Asset Allocation

The theory of optimal selection of portfolios was developed by Harry Markowitz in the 1950's. His work formalized the diversification principle in portfolio selection and, as mentioned above, earned him the 1990 Nobel prize for economics. We will discuss his model in more detail later. Here we give a brief description of the model and relate it to QPs.

Consider an investor who has a certain amount of money to be invested in a number of different securities (stocks, bonds, etc.) with random returns. For each security i, $i = 1, \ldots, n$, estimates of its expected return, μ_i, and variance, σ_i^2, are given. Furthermore, for any two securities i and j, their correlation coefficient ρ_{ij} is also assumed to be known. If we represent the proportion of the total funds invested in security i by x_i, one can compute the expected return and the variance of the resulting portfolio $x = (x_1, \ldots, x_n)$ as follows:

$$E[x] = x_1\mu_1 + \ldots + x_n\mu_n = \mu^T x,$$

and

$$Var[x] = \sum_{i,j} \rho_{ij}\sigma_i\sigma_j x_i x_j = x^T Q x$$

where $\rho_{ii} \equiv 1$, $Q_{ij} = \rho_{ij}\sigma_i\sigma_j$ for $i \neq j$, $Q_{ii} = \sigma_i^2$, and $\mu = (\mu_1, \ldots, \mu_n)$.

The portfolio vector x must satisfy $\sum_i x_i = 1$ and there may or may not be additional feasibility constraints. A feasible portfolio x is called *efficient* if it has the maximal expected return among all portfolios with the same variance, or alternatively, if it has the minimum variance among all portfolios that have at least a certain expected return. The collection of efficient portfolios form the *efficient frontier* of the portfolio universe.

Markowitz' *portfolio selection problem*, also called the *mean-variance optimization* (MVO) problem can be formulated in three different but equivalent ways. One formulation results in the problem of finding a minimum variance portfolio of the securities 1 to n that yields at least a target value of expected return (say b). Mathematically, this formulation produces a convex quadratic programming problem:

$$\begin{array}{lrcl} \min_x & x^T Q x & & \\ & e^T x & = & 1 \\ & \mu^T x & \geq & b \\ & x & \geq & 0, \end{array} \tag{1.13}$$

where e is a n-dimensional vector of ones. The first constraint indicates that the proportions x_i should sum to 1. The second constraint indicates that the expected return is no less than the target value and as we discussed above, the objective function

corresponds to the total variance of the portfolio. Nonnegativity constraints on x_i are introduced to disallow short sales (selling a security that you do not have). Note that the matrix Q is positive semidefinite since x^TQx, the variance of the portfolio, must be nonnegative for every portfolio (feasible or not) x.

The model (1.13) is rather versatile. For example, if short sales are permitted on some or all of the securities then this can be incorporated into the model simply by removing the nonnegativity constraint on the corresponding variables. If regulations or the investor's considerations limit the amount of investment in a subset of the securities, the model can be augmented with a linear constraint to reflect such a limit. In principle, any linear constraint can be added to the model without making it much harder to solve.

Asset allocation problems have an identical mathematical structure to portfolio selection problems. In these problems, the objective is not to choose a portfolio of stocks (or other securities), but to determine the optimal investment among a set of asset classes. Examples of these asset classes are large capitalization stocks, small capitalization stocks, foreign stocks, government bonds, corporate bonds, etc. Since there are many mutual funds focusing on each one of these different asset classes, one can conveniently invest in these asset classes by purchasing the corresponding mutual fund. After estimating the expected returns, variances, and covariances for different asset classes, one can formulate a QP identical to (1.13) and obtain efficient portfolios of these asset classes.

The formulation (1.13) we presented above makes several simplifying assumptions and much of the literature on asset allocation/portfolio selection focuses on solving this problem without some of these assumptions. We will address some of these variations and some other problems related to portfolio selection throughout the manuscript.

1.3.2 Pricing and Hedging of Options

We first start with a description of some of the well-known financial options. A *European call option* is a contract with the following conditions:

- At a prescribed time in the future, known as the *expiration date*, the *holder* of the option has the right, but not the obligation to
- purchase a prescribed asset, known as the *underlying*, for a
- prescribed amount, known as the *strike price* or *exercise price*.

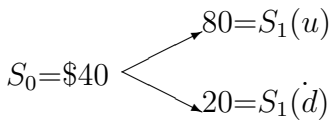
A *European put option* is like the call option, except that it gives the right to sell an asset. An *American call/put option* is like a European option, except that it may be exercised on or before the expiration date.

Since the payoff from an option depends on the value of the underlying security, its price is also related to the current value and expected behavior of this underlying security. To find the fair value of a given option, we need to solve a *pricing* problem and

this problem can often be solved using sophisticated mathematical techniques, provided that there is a good model for the stochastic behavior of the underlying security.

Option pricing problems are often solved using the following strategy: We try to determine a portfolio of assets with known prices which, if updated properly through time, will produce the same payoff as the option. Since the portfolio and the option will have the same eventual payoffs, we conclude that they must have the same value today (otherwise, there is *arbitrage*) and we can easily obtain the price of the option. A portfolio of other assets that produces the same payoff as a given financial instrument is called a *replicating portfolio* (or a *hedge*) for that instrument. Finding the right portfolio, of course, is not always easy and leads to a *replication* (or *hedging*) problem.

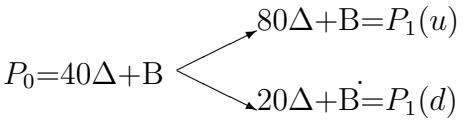
Let us consider a simple example to illustrate these ideas. Let us assume that one share of stock XYZ is currently valued at \$40. The price of XYZ a month from today is random: Assume that its value will either double or halve with equal probabilities.



Today, we purchase a European call option to buy one share of XYZ stock for \$50 a month from today. What is the fair price of this call option?

Let us assume that we can borrow or lend money with no interest between today and next month, and that we can buy or sell any amount of the XYZ stock without any commissions, etc. These are part of the "frictionless market" assumptions we will address later in the manuscript. Further assume that XYZ will not pay any dividends within the next month.

To solve the pricing problem, we consider the following hedging problem: Can we form a portfolio of the underlying stock (bought or sold) and cash (borrowed or lent) today, such that the payoff from the portfolio at the expiration date of the option will match the payoff of the option? Note that the option payoff will be \$30 if the price of the stock goes up and \$0 if it goes down. Say, this portfolio has Δ shares of XYZ and $\$B$ cash. This portfolio would be worth 40Δ+B today. Next month, payoffs for this portfolio will be:



Let us choose Δ and B such that

$$\begin{aligned} 80\Delta + B &= 30 \\ 20\Delta + B &= 0, \end{aligned}$$

so that the portfolio replicates the payoff of the option at the expiration date. This gives $\Delta = \frac{1}{2}$ and $B = -10$, which is the *hedge* we were looking for. This portfolio is worth $P_0 = 40\Delta + B =$\$10 today, therefore, the fair price of the option must also be \$10.

1.3.3 Risk Management

Risk is an inevitable consequence of productive activity. This is especially true for financial activities of companies and individuals where results of most decisions will be observed or realized in the future, in unpredictable circumstances. Since companies can not ignore such risks and can not insure themselves completely against risks, they have to manage it. This is a hard task even with the support of advanced mathematical techniques–poor risk management practices led to several spectacular failures in the financial industry during the last decade (e.g., Barings Bank, Long Term Capital Management, Orange County).

The development of a coherent risk management practice requires quantitative risk measures that adequately reflect the vulnerabilities of a company. Examples of these risk measures include portfolio variance as in the Markowitz MVO model, the Value-at-Risk (VaR) and the expected shortfall (also known as conditional VaR, or CVaR)). Furthermore, risk control techniques need to be developed and implemented to adapt to the rapid changes in the values of these risk measures. Government regulators already mandate that financial institutions control their holdings in certain ways and place margin requirements for "risky" positions.

Optimization problems encountered in financial risk management often take the following form: optimize a performance measure (such as expected investment return) subject to the usual operating constraints and the constraint that a particular risk measure for the companies financial holdings does not exceed a prescribed amount. Mathematically, we may have the following problem:

$$\begin{array}{rrcl} \max_x & \mu^T x & & \\ & \mathrm{RM}[x] & \leq & \gamma \\ & e^T x & = & 1 \\ & x & \geq & 0, \end{array} \tag{1.14}$$

As in the Markowitz MVO model, x_i represent the proportion of the total funds invested in security. The objective is the expected portfolio return and μ is the expected return vector for the different securities. $\mathrm{RM}[x]$ denotes the value of a particular risk measure for portfolio x and γ is the prescribed upper limit on this measure. Since $\mathrm{RM}[x]$ is generally a nonlinear function of x, (1.14) is a nonlinear programming problem. Alternatively, we may optimize the risk measure while requiring that expected return of the portfolio is at least a certain amount. This would produce a problem very similar to (1.13).

1.3.4 Asset Liability Management

How should a financial institution manage its assets and liabilities? A static mean-variance optimizing model such as the one we discussed for asset allocation fails to incorporate the multivariate nature of the liabilities faced by financial institutions.

Furthermore, it equally penalizes returns above the expected returns and shortfalls. A multi-period model that emphasizes the need to meet liabilities in each period for a finite (or possibly infinite) horizon is often required. Since liabilities and asset returns usually have random components, their optimal management requires tools of "Optimization under Uncertainty" and most notably, stochastic programming approaches.

Let L_t be the liability of the company in period t for $t = 1, \ldots, T$. Here, we assume that L_t's are random with known distributions. A typical problem to solve in asset/liability management is to determine which assets (and in what quantities) the company should hold in each period to maximize its expected wealth at the end of period T. We can further assume that the asset classes the company can choose from have random returns (again, with known distributions) denoted by R_{it} for asset class i in period t. Since the company can make the holding decisions for each period after observing the asset returns and liabilities in the previous periods, the resulting problem can be cast as a stochastic program with recourse:

$$\begin{array}{rrcl} \max_x & E[\sum_i x_{i,T}] & & \\ & \sum_i (1+R_{it}) x_{i,t-1} - \sum_i x_{i,t} & = & L_t,\ t = 1, \ldots, T \\ & x_{i,t} & \geq & 0 \ \ \forall i, t. \end{array} \tag{1.15}$$

The objective function represents the expected total wealth at the end of the last period. The constraints indicate that the surplus left after liability L_t is covered will be invested as follows: $x_{i,t}$ invested in asset i. In this formulation, $x_{0,t}$ are the fixed, and possibly nonzero initial positions in the different asset classes.

Chapter 2

Linear Programming: Theory and Algorithms

2.1 The Linear Programming Problem

One of the most common and fundamental optimization problems is the *linear programming* problem (LP), the problem of optimizing a linear objective function subject to linear equality and inequality constraints. A generic linear optimization problem has the following form:

$$(\mathcal{LOP}) \qquad \begin{array}{lrcl} \min_x & c^T x & & \\ & a_i^T x & = & b_i, i \in \mathcal{E} \\ & a_i^T x & \geq & b_i, i \in \mathcal{I}, \end{array} \tag{2.1}$$

where $\mathcal{E}$ and $\mathcal{I}$ are the index sets for linear equality and inequality constraints, respectively. For algorithmic purposes, it is often desirable to have the problems structured in a particular way. Since the development of the simplex method for LPs the following form has been a popular standard and is called the *standard form LP*:

$$(\mathcal{LP}) \qquad \begin{array}{lrcl} \min_x & c^T x & & \\ & Ax & = & b \\ & x & \geq & 0. \end{array} \tag{2.2}$$

Here $A \in \Re^{m \times n}$, $b \in \Re^m$, $c \in \Re^n$ are given, and $x \in \Re^n$ is the variable vector to be determined as the solution of the problem. The matrix A is assumed to have full row rank. This is done without loss of generality because if A does not have full row rank, the augmented matrix $[A|b]$ can be row reduced, which either reveals that the problem is infeasible or that one can continue with the reduced full-rank matrix.

This form is not restrictive: Inequalities (other than non-negativity) can be rewritten as equalities after the introduction of a so-called *slack* or *surplus* variable that is restricted to be nonnegative. For example,

$$\begin{array}{lrcrcl} \min & -x_1 & - & x_2 & & \\ & 2x_1 & + & x_2 & \leq & 12 \\ & x_1 & + & 2x_2 & \leq & 9 \\ & x_1, & & x_2 & \geq & 0 \end{array} \tag{2.3}$$

can be rewritten as

$$\begin{array}{lrcrcrcrcl} \min & -x_1 & - & x_2 & & & & & = & \\ & 2x_1 & + & x_2 & + & x_3 & & & = & 12 \\ & x_1 & + & 2x_2 & & & + & x_4 & = & 9 \\ & x_1, & & x_2, & & x_3, & & x_4 & \geq & 0. \end{array} \tag{2.4}$$

Variables that are not required to be nonnegative can be expressed as the difference of two new nonnegative variables. Simple transformations are available to rewrite any given LP in the standard form above. Therefore, in the rest of our theoretical and algorithmic discussion we assume that the LP is in the standard form.

Recall the following definitions from the introductory chapter: $\mathcal{LP}$ is said to be *feasible* if its constraints are consistent and it is called *unbounded* if there exists a sequence of feasible vectors $\{x^k\}$ such that $c^T x^k \to -\infty$. When we talk about a *solution* (without any qualifiers) to $\mathcal{LP}$ we mean any candidate vector $x \in \Re^n$. A *feasible solution* is one that satisfies the constraints, and an *optimal solution* is a vector x that satisfies the constraints and minimizes the objective value among all feasible vectors. When $\mathcal{LP}$ is feasible but not unbounded it has an optimal solution.

2.2 Duality

The most important questions we will address in this chapter are the following: How do we recognize an optimal solution and how do we find such solutions? Consider the standard form LP in (2.4) above. Here are a few alternative feasible solutions:

$$\begin{array}{rl} (x_1, x_2, x_3, x_4) = (0, \frac{9}{2}, \frac{15}{2}, 0) & \text{Objective value } = -\frac{9}{2} \\ (x_1, x_2, x_3, x_4) = (6, 0, 0, 3) & \text{Objective value } = -6 \\ (x_1, x_2, x_3, x_4) = (5, 2, 0, 0) & \text{Objective value } = -7 \end{array}$$

Since we are minimizing, the last solution is the best among the three feasible solutions we found, but is it the optimal solution? We can make such a claim if we can, somehow, show that there is no feasible solution with a smaller objective value.

Note that the constraints provide us some bounds on the value of the objective function. For example, for any feasible solution, we must have

$$-x_1 - x_2 \geq -2x_1 - x_2 - x_3 = -12$$

using the first constraint of the problem. The inequality above must hold for all feasible solutions since x_i's are all nonnegative and the coefficient of each variable on the LHS are at least as large as the coefficient of the corresponding variable on the RHS. We can do better using the second constraint:

$$-x_1 - x_2 \geq -x_1 - 2x_2 - x_4 = -9$$

and even better by adding a negative third of each constraint:

$$\begin{aligned} -x_1 - x_2 &\geq -x_1 - x_2 - \frac{1}{3}x_3 - \frac{1}{3}x_4 \\ &= -\frac{1}{3}(2x_1 + x_2 + x_3) - \frac{1}{3}(x_1 + 2x_2 + x_4) = -\frac{1}{3}(12 + 9) = -7. \end{aligned}$$

This last inequality indicates that for any feasible solution, the objective function value can not be smaller than -7. Since we already found a feasible solution achieving this bound, we conclude that this solution, namely $(x_1, x_2, x_3, x_4) = (5, 2, 0, 0)$ is an optimal solution of the problem.

This process illustrates the following strategy: If we find a feasible solution to the LP problem, and a bound on the optimal value of problem such that the bound and the objective value of the feasible solution coincide, then we can confidently recognize our feasible solution as an optimal solution. We will comment on this strategy shortly. Before that, though, we formalize our approach for finding a bound on the optimal objective value.

Our strategy was to find a linear combination of the constraints, say with multipliers y_1 and y_2 for the first and second constraint respectively, such that the combined coefficient of **each** variable forms a lower bound on the objective coefficient of that variable. In other words, we tried to choose y_1 and y_2 such that

$$y_1(2x_1+x_2+x_3)+y_2(x_1+2x_2+x_4) = (2y_1+y_2)x_1+(y_1+2y_2)x_2+y_1x_3+y_2x_4 \leq -x_1-x_2$$

or,

$$\begin{aligned} 2y_1 + y_2 &\leq -1 \\ y_1 + 2y_2 &\leq -1 \\ y_1 &\leq 0 \\ y_2 &\leq 0. \end{aligned}$$

Naturally, to obtain the best possible bound, we would like to find y_1 and y_2 that achieve the maximum combination of the right-hand-side values:

$$\max 12y_1 + 9y_2.$$

This process results in a linear programming problem that is strongly related to the LP we are solving. We want to

$$\begin{array}{lrcrcl} \max & 12y_1 & + & 9y_2 & & \\ & 2y_1 & + & y_2 & \leq & -1 \\ & y_1 & + & 2y_2 & \leq & -1 \\ & y_1 & & & \leq & 0 \\ & & & y_2 & \leq & 0. \end{array} \tag{2.5}$$

This problem is called the **dual** of the original problem we considered. The original LP in (2.2) is often called the *primal* problem. For a generic primal LP problem in standard form (2.2) the corresponding dual problem can be written as follows:

$$\begin{array}{lll} (\mathcal{LD}) & \max_y & b^T y \\ & & A^T y \leq c, \end{array} \tag{2.6}$$

where $y \in \Re^m$. Rewriting this problem with explicit **dual slacks**, we obtain the standard form dual linear programming problem:

$$\begin{array}{lll} (\mathcal{LD}) & \max_{y,s} & b^T y \\ & & A^T y + s = c \\ & & s \geq 0, \end{array} \tag{2.7}$$

where $s \in \Re^n$.

Next, we make some observations about the relationship between solutions of the primal and dual LPs. The objective value of any primal feasible solution is at least as large as the objective value of any feasible dual solution; this fact is known as the *weak duality theorem*:

Theorem 2.1 (Weak Duality Theorem) *Let x be any feasible solution to the primal LP (2.2) and y be any feasible solution to the dual LP (2.7). Then,*

$$c^T x \geq b^T y.$$

Proof:
Since $x \geq 0$ and $c - A^T y = s \geq 0$, the inner product of these two vectors must be nonnegative:

$$x^T s = s^T x = (c - A^T y)^T x = c^T x - y^T A x = c^T x - y^T b \geq 0.$$

□

The quantity $x^T s = c^T x - y^T b$ is often called the *duality gap*. The following three results are immediate consequences of the weak duality theorem:

Corollary 2.1 *If the primal LP is unbounded, then the dual LP must be infeasible.*

Corollary 2.2 *If the dual LP is unbounded, then the primal LP must be infeasible.*

Corollary 2.3 *If x is feasible for the primal LP, y is feasible for the dual LP, and $c^T x = b^T y$, then x must be optimal for the primal LP and y must be optimal for the dual LP.*

2.3 Optimality Conditions

The last corollary of the previous section identified a sufficient condition for optimality of a primal-dual pair of feasible solutions, namely that their objective values coincide. One natural question to ask is whether this is a necessary condition. The answer is yes, as we illustrate next.

Theorem 2.2 (Strong Duality Theorem) *If both the primal LP problem and the dual LP have feasible solutions than they both have optimal solutions and for any primal optimal solution x and dual optimal solution y we have that $c^T x = b^T y$.*

We will omit the (elementary) proof of this theorem since it requires some additional tools. The reader can find a proof of this result on most standard linear programming textbooks.

The strong duality theorem provides us with conditions to identify optimal solutions (called *optimality conditions*): $x \in \Re^n$ is an optimal solution of (2.2) if and only if

1. x is primal feasible: $Ax = b$, $x \geq 0$, and there exists a $y \in \Re^m$ and $s \in \Re^n$ such that
2. (y, s) is dual feasible: $A^T y + s = c$, $s \geq 0$, and
3. there is no duality gap: $c^T x = b^T y$.

Further analyzing the last condition above, we can obtain an alternative set of optimality conditions. Recall from the proof of the weak duality theorem that $c^T x - b^T y = (c - A^T y)^T x \geq 0$ for any feasible primal-dual pair of solutions, since it is given as an inner product of two nonnegative vectors. This inner product is 0 ($c^T x = b^T y$) if and only if the following statement holds: For each $i = 1, \ldots, n$, either x_i or $(c - A^T y)_i = s_i$ is zero. This equivalence is easy to see. All the terms in the summation on the RHS of the following equation are nonnegative:

$$0 = (c - A^T y)^T x = \sum_{i=1}^{n} (c - A^T y)_i x_i$$

Since the sum is zero, each term must be zero. Thus we found an alternative set of optimality conditions: $x \in \Re^n$ is an optimal solution of (2.2) if and only if

1. x is primal feasible: $Ax = b$, $x \geq 0$, and there exists a $y \in \Re^m$ and $s \in \Re^n$ such that
2. (y, s) is dual feasible: $A^T y \leq c$, $s \geq 0$, and
3. complementary slackness: for each $i = 1, \ldots, n$ we have $x_i s_i = 0$.

The best known (and most successful) methods for solving LPs are *interior-point methods* and the *simplex method*. We discuss the latter here and postpone our discussion of interior-point methods till we study quadratic programming problems.

2.4 The Simplex Method

To motivate our discussion of the simplex method, we consider the following example from the book *Introduction to Mathematical Programming* by R. Walker:

Example 2.1 *Farmer Jones has 100 acres of land to devote to wheat and corn and wishes to plan his planting to maximize the expected revenue. Jones has only $800 in capital to apply to planting the crops, and it costs $5 to plant an acre of wheat and $10 for an acre of corn. Their busy social schedule leaves the Jones family only 150 days of labor to devote to the crops. Two days will be required for each acre of wheat and one day for an acre of corn. If past experience indicates a return of $80 from each acre of wheat and $60 for each acre of corn, how many acres of each should be planted to maximize his revenue?*

Lettng variables x_1 and x_2 denote the number of acres used for wheat and corn respectively, we obtain the following formulation for Farmer Jones' problem:

$$\begin{array}{lrcrcl} \textit{Max } Z = & 80x_1 & + & 60x_2 & & \\ \textit{subject to:} & & & & & \\ & x_1 & + & x_2 & \leq & 100 \\ & 2x_1 & + & x_2 & \leq & 150 \\ & 5x_1 & + & 10x_2 & \leq & 800 \\ & x_1 & , & x_2 & \geq & 0. \end{array}$$

After we add slack variables to each of the functional constraints we obtain a representation of the problem in the standard form, suitable for the simplex method[1]*:*

$$\begin{array}{lrcrcrcrcrcl} \textit{Max } Z = & 80x_1 & + & 60x_2 & & & & & & & & \\ \textit{subject to:} & & & & & & & & & & & \\ & x_1 & + & x_2 & + & x_3 & & & & & = & 100 \\ & 2x_1 & + & x_2 & & & + & x_4 & & & = & 150 \\ & 5x_1 & + & 10x_2 & & & & & + & x_5 & = & 800 \\ & x_1 & , & x_2 & , & x_3 & , & x_4 & , & x_5 & \geq & 0. \end{array}$$

2.4.1 Basic Solutions

Let us consider a general LP problem in the following form:

$$\begin{array}{lrcl} \max & \mathbf{c}\,\mathbf{x} & & \\ & \mathbf{A}\mathbf{x} & \leq & \mathbf{b} \\ & \mathbf{x} & \geq & \mathbf{0}, \end{array}$$

[1]This representation is not exactly in the standard form since the objective is maximization rather than minimization. However, any maximization problem can be transformed into a minimization problem by multiplying the objective function by -1. Here, we avoid such a transformation to leave the objective function in its natural form–it should be straightforward to adapt the steps of the algorithm in the following discussion to address minimization problems.

where $\mathbf{A}$ is an $m \times n$ matrix with full row rank and $\mathbf{b}$ is an m-dimensional column vector and $\mathbf{c}$ is an n-dimensional row vector. The n-dimensional column vector $\mathbf{x}$ represents the variables of the problem. (In the Farmer Jones example we have $m = 3$ and $n = 5$.) Here is how we can represent these vectors and matrices:

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{bmatrix}, \ \mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix}, \ \mathbf{c} = \begin{bmatrix} c_1 & c_2 & \dots & c_n \end{bmatrix}, \mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}, \ \mathbf{0} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix},$$

Next, we add slack variables to each of the functional constraints to get the augmented form of the problem. Let $\mathbf{x}_s$ denote the vector of slack variables

$$\mathbf{x}_s = \begin{bmatrix} x_{n+1} \\ x_{n+2} \\ \vdots \\ x_{n+m} \end{bmatrix}$$

and let $\mathbf{I}$ denote the $m \times m$ identity matrix. Now, the constraints in the augmented form can be written as

$$\begin{bmatrix} \mathbf{A}, & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \mathbf{x_s} \end{bmatrix} = \mathbf{b}, \quad \begin{bmatrix} \mathbf{x} \\ \mathbf{x_s} \end{bmatrix} \geq \mathbf{0}. \tag{2.8}$$

To find basic solutions we consider partitions of the augmented matrix $[A, I]$:

$$\begin{bmatrix} \mathbf{A}, & \mathbf{I} \end{bmatrix} = \begin{bmatrix} \mathbf{B}, & \mathbf{N} \end{bmatrix},$$

where $\mathbf{B}$ is an $m \times m$ square matrix that consists of linearly independent columns of $[\mathbf{A}, \mathbf{I}]$. If we partition the variable vector $\begin{bmatrix} \mathbf{x} \\ \mathbf{x_s} \end{bmatrix}$ in the same way

$$\begin{bmatrix} \mathbf{x} \\ \mathbf{x_s} \end{bmatrix} = \begin{bmatrix} \mathbf{x_B} \\ \mathbf{x_N} \end{bmatrix},$$

we can rewrite the equality constraints in (2.8) as

$$\begin{bmatrix} \mathbf{B}, \mathbf{N} \end{bmatrix} \begin{bmatrix} \mathbf{x_B} \\ \mathbf{x_N} \end{bmatrix} = \mathbf{B}\mathbf{x_B} + \mathbf{N}\mathbf{x_N} = \mathbf{b},$$

or by multiplying both sides by B^{-1} from left,

$$\mathbf{x_B} + \mathbf{B^{-1}}\mathbf{N}\mathbf{x_N} = \mathbf{B^{-1}}\mathbf{b}.$$

So the following systems of equations are equivalent; any solution to the first will be a solution for the next two, and vice versa:

$$\begin{gathered}\left[\begin{array}{cc}\mathbf{A}, & \mathbf{I}\end{array}\right]\left[\begin{array}{c}\mathbf{x}\\ \mathbf{x_s}\end{array}\right]=\mathbf{b},\\ \mathbf{B}\mathbf{x_B}+\mathbf{N}\mathbf{x_N}=\mathbf{b}\\ \mathbf{x_B}+\mathbf{B^{-1}}\mathbf{N}\mathbf{x_N}=\mathbf{B^{-1}}\mathbf{b}\end{gathered}$$

Indeed, the linear systems in the second and third form are just re-representations of the first one with respect to a fixed matrix $\mathbf{B}$. An obvious solution to the last system (and therefore, for the other two) is $\mathbf{x_N}=\mathbf{0}$, $\mathbf{x_B}=\mathbf{B^{-1}}\mathbf{b}$. In fact, for any fixed values of the components of $\mathbf{x_N}$ we can obtain a solution by simply setting

$$\mathbf{x_B}=\mathbf{B^{-1}}\mathbf{b}-\mathbf{B^{-1}}\mathbf{N}\mathbf{x_N}.$$

The reader may want to think of $\mathbf{x_N}$ as the *independent* variables that we can choose freely, and once they are chosen, the *dependent* variables $\mathbf{x_B}$ are determined uniquely. We call a solution of the systems above a **basic solution** if it is of the form

$$\mathbf{x_N}=\mathbf{0},\ \mathbf{x_B}=\mathbf{B^{-1}}\mathbf{b},$$

for some **basis matrix B**. If in addition, $\mathbf{x_B}=\mathbf{B^{-1}}\mathbf{b}\geq\mathbf{0}$, the solution $\mathbf{x_B}=\mathbf{B^{-1}}\mathbf{b}$, $\mathbf{x_N}=\mathbf{0}$ is a **basic feasible solution** of the LP problem above. The variables $\mathbf{x_B}$ are called the **basic variables**, while $\mathbf{x_N}$ are the **nonbasic variables**.

The objective function $\mathbf{Z}=\mathbf{c}\ \mathbf{x}$ can be represented similarly using the basis partition. Let $\mathbf{c}=\left[\begin{array}{cc}\mathbf{c_B}, & \mathbf{c_N}\end{array}\right]$ be the represent the partition of the objective vector. Now, we have the following sequence of equivalent representations of the objective function equation:

$$\begin{gathered}\mathbf{Z}=\mathbf{c}\ \mathbf{x}\Leftrightarrow\mathbf{Z}-\mathbf{c}\ \mathbf{x}=\mathbf{0}\\ \mathbf{Z}-\left[\begin{array}{cc}\mathbf{c_B}, & \mathbf{c_N}\end{array}\right]\left[\begin{array}{c}\mathbf{x_B}\\ \mathbf{x_N}\end{array}\right]=\mathbf{0}\\ \mathbf{Z}-\mathbf{c_B}\ \mathbf{x_B}-\mathbf{c_N}\ \mathbf{x_N}=\mathbf{0}\\ \mathbf{Z}-\mathbf{c_B}\ (\mathbf{B^{-1}}\mathbf{b}-\mathbf{B^{-1}}\mathbf{N}\mathbf{x_N})-\mathbf{c_N}\ \mathbf{x_N}=\mathbf{0}\\ \mathbf{Z}-(\mathbf{c_N}-\mathbf{c_B}\mathbf{B^{-1}}\mathbf{N})\ \mathbf{x_N}=\mathbf{c_B}\mathbf{B^{-1}}\mathbf{b}\end{gathered}\tag{2.9}$$

Note that the last one of the list of equations above does not contain the basic variables, which is exactly what we want to be able to figure out the net effect of changing a nonbasic variable on the objective function.

A key observation is that when a linear programming problem has an optimal solution, it **must** have an optimal basic feasible solution. The significance of this result lies in the fact that when we are looking for a solution of a linear programming problem what we really need to check is the objective value of each basic solution. There are only finitely many of them, so this reduces our search space from an infinite space to a finite one.

2.4.2 Simplex Iterations

The simplex method solves a linear programming problem by moving from one basic feasible solution to another. Since one of these solutions is optimal, presumably, the method will eventually get there. But first, it has to start at a basic feasible solution. For the Farmer Jones problem, this is a trivial task, choosing

$$\mathbf{B} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \mathbf{x_B} = \begin{bmatrix} x_3 \\ x_4 \\ x_5 \end{bmatrix}, \mathbf{N} = \begin{bmatrix} 1 & 1 \\ 2 & 1 \\ 5 & 10 \end{bmatrix}, \mathbf{x_N} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

we get an initial basic feasible solution (BFS) with $\mathbf{x_B} = \mathbf{B^{-1}b} = [100, 150, 800]^T$. The objective value for this BFS is $80 \cdot 0 + 60 \cdot 0 = 0$.

We first need to determine whether this solution is optimal. We observe that both x_1 and x_2 would improve the objective value if they were introduced to the basis. Why? Our initial basic feasible solution has $x_1 = x_2 = 0$ and we can get other solutions by varying the value of one of these two nonbasic variables. Furthermore, we know that we could increase either of these two nonbasic variables and balance the equality conditions by changing the basic variables x_3, x_4, and x_5. None of these variables appear in the objective row. As a matter of fact, these variables do not appear anywhere other than the row where they are the basic variable, so changes in the values of basic variables do not effect the objective, we only have to look at the coefficient of the nonbasic variable we would increase to see what effect this would have on the objective value. The rate of improvement in the objective value for x_1 is 80 and for x_2 this rate is only 60. We pick the variable x_1 to enter the basis since it has a faster rate of improvement.

Next, we need to find a variable to leave the basis, because the basis holds only 3 variables[2]. We know that nonbasic variables of a basic solution, so we need to determine how much to increase x_1 so that one of the current basic variables becomes zero. The important issue here is to maintain the non-negativity of all basic variables. As we increase x_1, all current basic variables will decrease since x_1 has positive coefficients in each row[3]. We guarantee the non-negativity of the basic variables of the next iteration by using the ratio test. We observe that

$$\begin{array}{lcl} \text{increasing } x_1 \text{ beyond } 100/1{=}100 & \Rightarrow & x_3 < 0, \\ \text{increasing } x_1 \text{ beyond } 150/2{=}75 & \Rightarrow & x_4 < 0, \\ \text{increasing } x_1 \text{ beyond } 800/5{=}160 & \Rightarrow & x_5 < 0, \end{array}$$

[2]3 is the number of equations here. For a general LP, the size of the basis will be equal to the number of equations in the standard form representation of the problem.

[3]If x_1 had a zero coefficient in a particular row, then increasing it would not effect the basic variable in that row. If, x_1 had a negative coefficient in a row, then as x_1 was being increased the basic variable of that row would need to be increased to maintain the equality in that row; but then we would not worry about that basic variable becoming negative.

so we should not increase x_1 more than $\min\{100, 75, 800\} = 75$, if we want to maintain the non-negativity of the basic variables. On the other hand if we increase x_1 exactly by 75, x_4 will become zero, and we can choose it as the leaving variable.

Now we have a new basis: $\{x_3, x_1, x_5\}$. For this basis we have the following basic feasible solution:

$$\mathbf{B} = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 2 & 0 \\ 0 & 5 & 1 \end{bmatrix}, \mathbf{x}_B = \begin{bmatrix} x_3 \\ x_1 \\ x_5 \end{bmatrix} = \mathbf{B^{-1}b} = \begin{bmatrix} 1 & -1/2 & 0 \\ 0 & 1/2 & 0 \\ 0 & -5/2 & 1 \end{bmatrix} \begin{bmatrix} 100 \\ 150 \\ 800 \end{bmatrix} = \begin{bmatrix} 25 \\ 75 \\ 425 \end{bmatrix},$$

$$\mathbf{N} = \begin{bmatrix} 1 & 0 \\ 1 & 1 \\ 10 & 0 \end{bmatrix}, \mathbf{x_N} = \begin{bmatrix} x_2 \\ x_4 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}.$$

As always, after finding a new feasible solution, we ask ourselves 'Is this the optimal solution, or can we still improve it?'. Answering that question was easy when we started, because none of the basic variables were in the objective function. Now that we have introduced x_1 into the basis, the situation is more complicated. If we now decide to increase x_2, the objective row coefficient of x_2 would not necessarily tell us how much the objective value change per unit change in x_2, because changing x_2 would necessitate changing x_1, a basic variable that appears in the objective row. It may happen that, increasing x_2 by 1 unit does not increase the objective value by 60 units, because x_1 may need to be decreased, pulling down the objective function. Indeed, it may happen that increasing x_2 actually decreases the objective value, while it had a positive coefficient in the objective function at the beginning. So, what do we do? We could do what we did with the initial basic solution **if** x_1 did not appear in the objective row and the rows where it is not the basic variable. But this is not very hard to achieve: we can use the row where x_1 is the basic variable (in this case the second row) to solve for x_1 in terms of the nonbasic variables and then substitute this expression for x_1 in the objective row and other equations. So, the second equation

$$2x_1 + x_2 + x_4 = 150$$

would give us:

$$x_1 = 75 - \frac{1}{2}x_2 - \frac{1}{2}x_4.$$

Substituting this value in the objective function we get:

$$Z = 80x_1 + 60x_2 = 80(75 - \frac{1}{2}x_2 - \frac{1}{2}x_4) + 60x_2 = 6000 + 20x_2 - 40x_4.$$

Continuing the substitution we get the following representation of the original Farmer Jones problem:

Maximize Z
subject to:

$$
\begin{array}{lrcrcrcrcrcr}
Z & -20x_2 & + & 40x_4 & & & & & & = & 6000 \\
 & \frac{1}{2}x_2 & - & \frac{1}{2}x_4 & + & x_3 & & & & = & 25 \\
 & \frac{1}{2}x_2 & + & \frac{1}{2}x_4 & & & + & x_1 & & = & 75 \\
 & \frac{15}{2}x_2 & - & \frac{5}{2}x_4 & & & & & + \; x_5 & = & 425 \\
 & x_2 & , & x_4 & , & x_3 & , & x_1 & , \; x_5 & \geq & 0.
\end{array}
$$

We now achieved what we wanted to; once again, the objective row is free of basic variables and basic variables only appear with a coefficient of 1 in the row that they are basic. This representation looks exactly like the initial system. Therefore, we now can tell how a change in the nonbasic variables would effect the objective function: increasing x_2 by 1 unit will increase the objective function by 20 units (not 60!) and increasing x_4 by 1 unit will decrease the objective function by 40 units.

Now that we represented the problem in a form identical to the original, we can repeat what we did before, until we find a representation that gives the optimal solution. If we repeat the steps of the simplex method , we find that x_2 will be introduced into the basis next, and the leaving variable will be x_3. If we solve for x_1 using the first equation and substitute for it in the remaining ones, we get the following representation:

Maximize Z
subject to:

$$
\begin{array}{lrcrcrcrcrcr}
Z & +40x_3 & + & 20x_4 & & & & & & = & 7000 \\
 & 2x_3 & - & x_4 & + & x_2 & & & & = & 50 \\
 & -x_3 & + & x_4 & & & + & x_1 & & = & 50 \\
 & -15x_3 & + & 5x_4 & & & & & + \; x_5 & = & 50 \\
 & x_3 & , & x_4 & , & x_2 & , & x_1 & , \; x_5 & \geq & 0.
\end{array}
$$

Once again, notice that this representation is very similar to the tableau we got at the end of the previous section. The basis and the basic solution that corresponds to the system above is:

$$
\mathbf{B} = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 2 & 0 \\ 10 & 5 & 1 \end{bmatrix}, \mathbf{x_B} = \begin{bmatrix} x_2 \\ x_1 \\ x_5 \end{bmatrix} = B^{-1}b = \begin{bmatrix} 2 & -1 & 0 \\ -1 & 1 & 0 \\ -15 & 5 & 1 \end{bmatrix} \begin{bmatrix} 100 \\ 150 \\ 800 \end{bmatrix} = \begin{bmatrix} 50 \\ 50 \\ 50 \end{bmatrix},
$$

$$
\mathbf{N} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix}, \mathbf{x_N} = \begin{bmatrix} x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}.
$$

At this point we can conclude that this basic solution is the optimal solution. Let us try to understand why this last basic solution is optimal. From our discussion at the beginning of this section, recall that any feasible solution to a linear programming

problem in the standard form can be expressed relative to a basis matrix B by $x_B = B^{-1}b - B^{-1}Nx_N$, where $x_N \geq 0$ because of the non-negativity restrictions. Then, if we decide to use the basis B above to express the system, any feasible solution to Farmer Jones' problem can be found by choosing certain nonnegative values for x_3 and x_4 and setting

$$\mathbf{x_B} = \begin{bmatrix} x_2 \\ x_1 \\ x_5 \end{bmatrix} = \mathbf{B^{-1}b - B^{-1}Nx_N} = \begin{bmatrix} 50 \\ 50 \\ 50 \end{bmatrix} - \begin{bmatrix} 2 & -1 & 0 \\ -1 & 1 & 0 \\ -15 & 5 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x_3 \\ x_4 \end{bmatrix}.$$

Since x_3 and x_4 are currently zero and must stay nonnegative, the only way to obtain a different feasible solution is to increase one (or both) of these two variables. However, if we look at the objective function in this last representation, we see that increasing either of these variables would only decrease the objective function. Putting these two observations together, we see that any other feasible solution to this problem must have an objective value smaller than the current basic feasible solution, which means that the current solution is the best, the optimal.

2.4.3 The Tableau Form of the Simplex Method

In most linear programming textbooks, the simplex method is described using *tableaus* that summarize the information in the different representations of the problem we saw above. Since the reader will likely encounter simplex tableaus if s/he studies optimization problems, we include a brief discussion for the purpose of completeness. To study the tableau form of the simplex method, we will recall the Farmer Jones example. We begin by rewriting the objective row as

$$Z - 80\,x_1 - 60\,x_2 = 0$$

and represent this system using the following tableau:

		⇓					
	Basic var.	x_1	x_2	x_3	x_4	x_5	
	Z	-80	-60	0	0	0	0
	x_3	1	1	1	0	0	100
⇐	x_4	2*	1	0	1	0	150
	x_5	5	10	0	0	1	800

This tableau is often called the *simplex tableau*. The columns labeled by each variable contain the coefficients of that variable in each equation, including the objective row equation. The leftmost column is used to keep track of the basic variable in each row. The arrows and the asterisk will be explained below.

Step 0. *Form the initial tableau.*

Once we have formed this tableau we look for an **entering variable**, i.e., a variable that has a negative coefficient in the objective row and will improve the objective function if it is introduced into the basis. In this case, two of the variables, namely x_1 and x_2, have negative objective row coefficients. Since x_1 has the most negative coefficient we will pick that one (this is indicated by the arrow pointing down on x_1), but in principle any variable with a negative coefficient in the objective row can be chosen to enter the basis.

Step 1. *Find a variable with a negative coefficient in the first row (the objective row). If all variables have nonnegative coefficients in the objective row, STOP, the current tableau is optimal.*

After we choose x_1 as the entering variable, we need to determine a **leaving variable** (sometimes called a departing variable). The leaving variable is found by performing a *ratio test*. In the ratio test, one looks at the column that corresponds to the entering variable, and for each **positive** entry in that column computes the ratio of that positive number to the right hand side value in that row. The minimum of these ratios tells us how much we can increase our entering variable without making any of the other variables negative. The basic variable in the row that gives the minimum ratio becomes the leaving variable. In the tableau above the column for the entering variable, the column for the right-hand-side values, and the ratios of corresponding entries are

$$\begin{matrix} x_1 \\ \left[\begin{array}{c} 1 \\ 2 \\ 5 \end{array}\right] \end{matrix}, \quad \begin{matrix} \text{RHS} \\ \left[\begin{array}{c} 100 \\ 150 \\ 800 \end{array}\right] \end{matrix}, \quad \begin{matrix} \text{ratio} \\ 100/1 \\ 150/2 \\ 800/5 \end{matrix} \quad , \min\{\frac{100}{1}, \frac{150^*}{2}, \frac{800}{5}\} = 75,$$

and therefore x_4, the basic variable in the second row, is chosen as the leaving variable, as indicated by the left arrow in the tableau.

One important issue here is that, we only look at the positive entries in the column when we perform the ratio test. Notice that if some of these entries were negative, then increasing the entering variable would only increase the basic variable in those rows, and would not force them to be negative, therefore we need not worry about those entries. Now, if all of the entries in a column for an entering variable turn out to be zero or negative, then we conclude that the problem must be *unbounded*; we can increase the entering variable (and the objective value) indefinitely, the equalities can be balanced by *increasing* the basic variables appropriately, and none of the nonnegativity constraints will be violated.

Step 2. *Consider the column picked in Step 1. For each positive entry in this column, calculate the ratio of the right-hand-side value to that entry. Find the row that gives minimum such ratio and choose the basic variable in that row as the leaving variable. If all the entries in the column are zero or negative, STOP, the problem is unbounded.*

Before proceeding to the next iteration, we need to update the tableau to reflect the changes in the set of basic variables. For this purpose, we choose a **pivot element**, which is the entry in the tableau that lies in the intersection of the column for the entering variable (the *pivot column*), and the row for the leaving variable (the *pivot row*). In the tableau above, the pivot element is the number 2, marked with an asterisk. The next job is **pivoting**. When we pivot, we aim to get the number 1 in the position of the pivot element (which can be achieved by dividing the entries in the pivot row by the pivot element), and zeros elsewhere in the pivot column (which can be achieved by adding suitable multiples of the pivot row to the other rows, including the objective row). All these operations are row operations on the matrix that consists of the numbers in the tableau, and what we are doing is essentially Gaussian elimination on the pivot column. Pivoting on the tableau above yields:

			⇓				
	Basic var.	x_1	x_2	x_3	x_4	x_5	
	Z	0	-20	0	40	0	6000
⇐	x_3	0	$1/2^*$	1	-1/2	0	25
	x_1	1	1/2	0	1/2	0	75
	x_5	0	15/2	0	-5/2	1	425

Step 3. *Find the entry (the pivot element) in the intersection of the column picked in Step 1 (the pivot column) and the row picked in Step 2 (the pivot row). Pivot on that entry, i.e., divide all the entries in the pivot row by the pivot element, add appropriate multiples of the pivot row to the others in order to get zeros in other components of the pivot column. Go to Step 1.*

If we repeat the steps of the simplex method, this time working with the new tableau, we first identify x_2 as the only candidate to enter the basis. Next, we do the ratio test:

$$\min\{\frac{25}{1/2}, \frac{75}{1/2}, \frac{425}{15/2}\} = 50,$$

so x_3 leaves the basis. Now, pivoting produces the **optimal tableau**:

Basic var.	x_1	x_2	x_3	x_4	x_5	
Z	0	0	40	20	0	7000
x_2	0	1	2	-1	0	50
x_1	1	0	-1	1	0	50
x_5	0	0	-15	5	1	50

This solution is optimal since all the coefficients in the objective row are nonnegative.

2.5 Exercises

1. Consider the following linear programming problem:

$$\begin{aligned} \max\ & 4x_1 + 3x_2 & & \\ & 3x_1 + x_2 & \leq & 9 \\ & 3x_1 + 2x_2 & \leq & 10 \\ & x_1 + x_2 & \leq & 4 \\ & x_1, x_2 \geq 0. & & \end{aligned}$$

   First, transform this problem into the "standard form". How many basic solutions does the standard form problem have? How many basic feasible solutions? What are the basic feasible solutions and what are the extreme points of the feasible region?

2. We say that two linear programming problems are equivalent if one can be obtained from the other by (i) multiplying the objective function by -1 and changing it from min to max, or max to min, and/or (ii) multiplying some or all constraints by -1. For example, $\{\min \mathtt{c^T x} : \mathtt{s.t. Ax} \geq \mathtt{b}\}$ and $\{\max -\mathtt{c^T x} : \mathtt{s.t.} -\mathtt{Ax} \leq -\mathtt{b}\}$ are equivalent problems. Find a linear program which is equivalent to its own dual.

Chapter 3

LP Models and Tools in Finance

3.1 Derivative Securities and The Fundamental Theorem of Asset Pricing

One of the most widely studied problems in financial mathematics is the pricing of *derivative securities*, also known as *contingent claims*. These are securities whose price depend on the value of another *underlying security*. Financial *options* are the most common examples of derivative securities. For example, a European call option on a particular underlying security gives the holder the right to purchase this underlying security for a prescribed amount (called the *strike price*) at a prescribed time in the future, known as the *expiration* or *exercise date*. The exercise date is also known as the *maturity date* of the derivative security. Recall the definitions of European put options as well as American call and put options from Section 1.3.2.

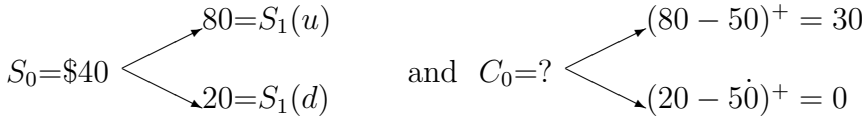
Options are used mainly for two purposes: speculation and hedging. By speculating on the direction of the future price movements of the underlying security, investors can take (bare) positions in options on this security. Since options are often much cheaper than their underlying security, this gamble results in much larger earnings if the price movements happen in the expected direction compared to what one might earn by taking a similar position in the underlying. Of course, if one guesses the direction of the price movements incorrectly, the losses are also much more severe.

Hedging refers to the reduction of risk in an investor's overall position by forming a suitable portfolio of the underlying and an option, or multiple options, on it. For example, if an investor holds a share of XYZ and is concerned that the price of this security may fall significantly, she can purchase a put option on XYZ and protect herself against price levels below a certain threshold (the strike price of the put option).

Recall the option example in the simple one-period binomial model of Section 1.3.2. Below, we summarize some of the information from that example:

We considered the share price of XYZ stock which is currently valued at \$40. A month from today, we expect the share price of XYZ to either double or halve, with

equal probabilities. We also considered a European call option on XYZ with a strike price of \$50 which will expire a month from today. We assumed that interest rates for cash borrowing or lending are zero and that any amount of XYZ shares can be bought or sold with no commission.



We obtained a fair price of \$10 for the option using a replication strategy and the no-arbitrage principle which essentially means that two portfolios of securities that have identical future payoffs under all possible realizations of the random states must have the same value today. In the example, the first "portfolio" is the option while the second one is the portfolio of $\frac{1}{2}$ share of XYZ and -\$10 in cash. Since we knew the current value of the second portfolio, we could deduce the fair price of the option. Let us give a formal definition of arbitrage:

Definition 3.1 *An* arbitrage *is a trading strategy*

- *that has a positive initial cash flow and has no risk of a loss later (type A), or*
- *that requires no initial cash input, has no risk of a loss, and a positive probability of making profits in the future (type B).*

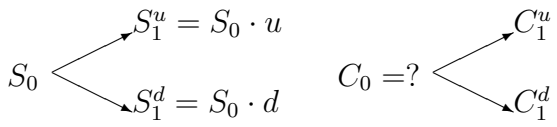
In the example, any price other than \$10 for the call option would lead to a type A arbitrage–guaranteed profits at the initial time point and no future obligations. We do not need to have a guarantee of profits for type B arbitrage–all we need is a guarantee of no loss, and a positive probability of a gain. Prices adjust quickly so that arbitrage opportunities can not persist in the markets. Therefore, in mathematical arguments it is often assumed that arbitrage opportunities do not exist.

3.1.1 Replication

Consider the following question: Given an option on a particular underlying security, can one form a portfolio of the underlying security (bought or sold) and cash (borrowed or lent) today, such that the payoff from the portfolio at the expiration date of the option will match the payoff of the option? In other words, can we replicate the option using a portfolio of the underlying and cash? This is the problem we formulated and solved to determine the fair price of the option in our example above.

Let us work in a slightly more general setting. Let S_0 be the current price of the underlying security and assume that there are two possible outcomes at the end of the period: $S_1^u = S_0 \cdot u$ and $S_1^d = S_0 \cdot d$. (Assume $u > d$.) We also assume that there is a fixed interest paid on cash borrowed or lent at rate r for the given period. Let $R = 1 + r$.

Finally, we consider a derivative security which has payoffs of C_1^u and C_1^d in the up and down states respectively:



To price the derivative security, we will try to replicate it. For replication consider a portfolio of Δ shares of the underlying and $\$B$ cash. For what values of Δ and B does this portfolio has the same payoffs at the expiration date as the derivative security?

We need to solve the following simple system of equations:

$$\begin{aligned} \Delta S_0 \cdot u + BR &= C_1^u \\ \Delta S_0 \cdot d + BR &= C_1^d. \end{aligned}$$

We obtain:

$$\begin{aligned} \Delta &= \frac{C_1^u - C_1^d}{S_0(u-d)} \\ B &= \frac{uC_1^d - dC_1^u}{R(u-d)} \end{aligned}$$

This portfolio is worth $S_0\Delta + B$ today, which should be the price of the derivative security as well:

$$\begin{aligned} C_0 &= \frac{C_1^u - C_1^d}{u-d} + \frac{uC_1^d - dC_1^u}{R(u-d)} \\ &= \frac{1}{R}\left[\frac{R-d}{u-d}C_1^u + \frac{u-R}{u-d}C_1^d\right]. \end{aligned}$$

3.1.2 Risk-Neutral Probabilities

Let

$$p_u = \frac{R-d}{u-d} \quad \text{and} \quad p_d = \frac{u-R}{u-d}.$$

Note that we must have $d < R < u$ to avoid arbitrage opportunities (see Exercises). An immediate consequence of this observation is that both $p_u > 0$ and $p_d > 0$. Noting also that $p_u + p_d = 1$ one can interpret p_u and p_d as probabilities. In fact, these are the so-called *risk-neutral probabilities* (RNPs) of up and down states, respectively. Note that they are completely independent from the actual probabilities of these states.

The price of any derivative security can now be calculated as the present value of the expected value of its future payoffs where the expected value is taken using the risk-neutral probabilities.

In our example above $u = 2$, $d = \frac{1}{2}$ and $r = 0$ so that $R = 1$. Therefore:

$$p_u = \frac{1-1/2}{2-1/2} = \frac{1}{3} \quad \text{and} \quad p_d = \frac{2-1}{2-1/2} = \frac{2}{3}.$$

As a result, we have

$$\begin{aligned} S_0 = 40 &= \frac{1}{R}(p_u S_1^u + p_d S_1^d) = \frac{1}{3}80 + \frac{2}{3}20, \\ C_0 = 10 &= \frac{1}{R}(p_u C_1^u + p_d C_1^d) = \frac{1}{3}30 + \frac{2}{3}0, \end{aligned}$$

as expected. Using risk neutral probabilities we can also price other derivative securities on the XYZ stock. For example, consider a European put option on the XYZ stock struck at \$60 (this is another way to say "with a strike price of \$60") and with the same expiration date as the call of the example.

$$P_0 = ? \begin{cases} \nearrow P_1^u = \max\{0, 60 - 80\} = 0 \\ \searrow P_1^d = \max\{0, 60 - 20\} = 40 \end{cases}$$

We can easily compute:

$$P_0 = \frac{1}{R}(p_u P_1^u + p_d P_1^d) = \frac{1}{3}0 + \frac{2}{3}40 = \frac{80}{3},$$

without needing to replicate the option again.

Next we move from our binomial setting to a more general setting and let

$$\Omega = \{\omega_1, \omega_2, \ldots, \omega_m\} \tag{3.1}$$

be the (finite) set of possible future "state"s. For example, these can be the possible prices for a security in a future date.

For securities S^i for $i = 0 \ldots n$, let $S_1^i(\omega_j)$ denote the price of this security in state ω_j at time 1. Also let S_0^i denote the current (time 0) prices of each one of these securities. We use $i = 0$ for the "riskless" security that pays the interest rate $r \geq 0$ between time 0 and time 1. It is convenient to assume that $S_0^0 = 1$ and that $S_1^0(\omega_j) = R := 1 + r, \forall j$.

Definition 3.2 *A risk-neutral probability measure is a vector of positive numbers* $(p_1, p_2, \ldots, p_m)$ *such that*

$$\sum_{j=1}^{m} p_j = 1$$

and for every security S^i *for* $i = 0, \ldots, n$,

$$S_0^i = \frac{1}{R}\left(\sum_{j=1}^{m} p_j S_1^i(\omega_j)\right) = \frac{1}{R}\hat{\mathbf{E}}[S_1^i].$$

Above, $\hat{\mathbf{E}}[S]$ denotes the expected value of the random variable S under the probability distribution $(p_1, p_2, \ldots, p_m)$.

We complete this section by stating and proving the first fundamental theorem of asset pricing. For finite Ω this proof is a simple exercise in linear programming duality that also utilizes the following well-known result of Goldman and Tucker on the existence of strictly complementary optimal solutions of LPs:

Theorem 3.1 *When both the primal and dual linear programming problems*

$$(\mathcal{LP}) \qquad \begin{array}{rcl} \min_x \quad c^T x & & \\ Ax & = & b \\ x & \geq & 0. \end{array} \tag{3.2}$$

and

$$(\mathcal{LD}) \qquad \begin{array}{rcl} \max_y \quad b^T y & & \\ A^T y & \leq & c, \end{array} \tag{3.3}$$

have feasible solutions, they have optimal solutions satisfying strict complementarity, i.e., there exists x^* and y^* optimal for the respective problems such that

$$x^* + (c - A^T y^*) > 0.$$

Now, we are ready to prove the following theorem:

Theorem 3.2 (The First Fundamental Theorem of Asset Pricing) *A risk-neutral probability measure exists if and only if there is no arbitrage.*

Proof:
We assume that the state space Ω is finite and is given by (3.1). Given the current prices S_0^i, and future prices $S_1^i(\omega_j)$ in each one of the states ω_j of securities 0 to n, consider the following linear programming problem with variables x_i, for $i = 0, \ldots, n$:

$$\begin{array}{rcl} \min_x \quad \sum_{i=0}^n S_0^i x_i & & \\ \sum_{i=0}^n S_1^i(\omega_j) x_i & \geq & 0, \quad j = 1, \ldots, m. \end{array} \tag{3.4}$$

Note that, type-A arbitrage corresponds to a feasible solution to this LP with a negative objective value. Since $x_i \equiv 0$ is always a feasible solution for this problem, the optimal objective value is always non-positive. Furthermore, since all the constraints are homogeneous, if there exists a feasible solution such that $S_0^i x_i < 0$ (this corresponds to type-A arbitrage), the problem is unbounded. In other words, there is no type-A arbitrage if and only if the optimal objective value of this LP is 0.

Suppose that there is no type-A arbitrage. Then, there is no type-B arbitrage if and only if all constraints are tight for all optimal solutions of (3.4). Note that these solutions must have objective value 0.

Consider the dual of (3.4):

$$\begin{array}{rcl} \max_p \quad \sum_{j=1}^m 0 p_j & & \\ \sum_{j=1}^m S_1^i(\omega_j) p_j & = & S_0^i, \quad i = 0, \ldots, n, \\ p_j & \geq & 0, \quad j = 1, \ldots, m. \end{array} \tag{3.5}$$

Since the dual has a constant (0) objective function, any dual feasible solution is also dual optimal.

When there is no type-A arbitrage (3.4) has an optimal solution, and the Strong Duality Theorem indicates that the dual must have a feasible solution. If there is no type-B arbitrage also, Goldman and Tucker's theorem indicates that, there exists a feasible (and therefore optimal) dual solution p^* such that $p^* > 0$ (from strict complementarity with tight primal constraints $\sum_{i=1}^{n} S_1^i(\omega_j)x_i \geq 0$). From the dual constraint corresponding to $i = 0$, we have that $\sum_{j=1}^{m} p_j^* = \frac{1}{R}$. Multiplying p^* by R one obtains a risk-neutral probability distribution. Therefore, "no arbitrage" assumption implies the existence of RNPs.

The converse direction is proved in an identical manner. The existence of a RNP measure implies that (3.5) is feasible, and therefore its dual, (3.4) must be bounded, which implies that there is no type-A arbitrage. Furthermore, since we have a strictly feasible (and optimal) dual solution, any optimal solution of the primal must have tight constraints, indicating that there is no type-B arbitrage. □

3.2 Arbitrage Detection Using Linear Programming

The linear programming problems (3.4) and (3.4) we formulated for the proof of Theorem 3.2 can naturally be used for detection of arbitrage opportunities. However, as we discussed above, this argument works only for finite state spaces. In this section, we discuss how LP formulations can be used to detect arbitrage opportunities without limiting consideration to finite state spaces. The price we pay for this flexibility is the restriction on the selection of the securities: we only consider the prices of a set of derivative securities written on the same underlying with same maturity. This discussion is based on [7].

Consider an underlying security whose current (time 0) price is given by S_0 and its (random) price at time 1 is denoted by S_1. Consider n derivative securities written on this security that mature at time 1, and have piecewise linear payoff functions with a single breakpoint. The obvious motivation for such a consideration is the the collection of calls and puts written on this security. Let us denote these piecewise linear payoff function of i-th derivative security with $\Psi_i(S_1)$ and its breakpoint with K_i. If, for example, the i-th derivative security were a European call with strike price K_i, we would have $\Psi_i(S_1) = (S_1 - K_i)^+$. We assume that K_i's are in increasing order, without loss of generality. Finally, let S_0^i denote the current price of the i-th derivative security.

Consider a portfolio $x = (x_1, \ldots, x_n)$ of the derivative securities 1 to n and let $\Psi^x(S_1)$ denote the payoff function of the portfolio:

$$\Psi^x(S_1) \quad = \quad \sum_{i=1}^{n} \Psi_i(S_1)x_i. \tag{3.6}$$

The cost of the portfolio x is given by

$$\sum_{i=1}^{n} S_0^i x_i. \tag{3.7}$$

To determine whether there exists a static arbitrage opportunity in the current prices S_0^i, we consider the following problem: What is the smallest cost portfolio of the derivative securities 1 to n whose payoff function $\Psi^x(S_1)$ is nonnegative for all $S_1 \in [0, \infty)$? Non-negativity of $\Psi^x(S_1)$ corresponds to "no future obligations". If the minimum initial cost of such a portfolio is negative, then we have type-A arbitrage.

Since all $\Psi_i(S_1)$'s are piecewise linear, so is $\Psi^x(S_1)$ with breakpoints in K_1 through K_n. Note that a piecewise linear function is nonnegative over $[0, \infty)$ if and only if it is nonnegative at 0 and all the breakpoints and if the slope of the function is nonnegative to the right of the largest breakpoint. From this observation, it easily follows that $\Psi^x(S_1)$ is nonnegative for all non-negative values of S_1 if and only if

1. $\Psi^x(0) \geq 0$,
2. $\Psi^x(K_j) \geq 0, \ \forall j$,
3. and $[(\Psi^x)'_+(K_n)] \geq 0$.

Now consider the following linear programming problem:

$$\begin{array}{lrcl} \min_x & \sum_{i=0}^n S_0^i x_i & & \\ & \sum_{i=0}^n \Psi_i(0) x_i & \geq & 0 \\ & \sum_{i=0}^n \Psi_i(K_j) x_i & \geq & 0, \quad j = 1, \ldots, n \\ & \sum_{i=0}^n (\Psi_i(K_n + 1) - \Psi_i(K_n)) x_i & \geq & 0 \end{array} \tag{3.8}$$

Since all $\Psi_i(S_1)$'s are piecewise linear, the quantity $\Psi_i(K_n + 1) - \Psi_i(K_n)$ gives the right-derivative of $\Psi_i(S_1)$ at K_n and the expression in the last constraint is the right derivative of $\Psi^x(S_1)$ at K_n. The following observation follows from our arguments above:

Proposition 3.1 *There is no type-A arbitrage in prices S_0^i if and only if the optimal objective value of (3.8) is zero.*

Similar to the previous section, we have the following result:

Proposition 3.2 *Suppose that there are no type-A arbitrage opportunities in prices S_0^i. Then, there are no type-B arbitrage opportunities if and only if the dual of the problem (3.8) has a strictly feasible solution.*

Proof:
Left as an exercise.

Next, we focus on the case where the derivative securities under consideration are European call options with strikes at K_i for $i = 1, \ldots, n$, so that $\Psi_i(S_1) = (S_1 - K_i)^+$ and

$$\Psi_i(K_j) = (K_j - K_i)^+.$$

In this case, the problem (3.8) reduces to the following problem:

$$\begin{array}{rcl} \min_x \quad c^T x & & \\ Ax & \geq & 0, \end{array} \tag{3.9}$$

where $c = [S_0^1, \ \ldots, S_0^n]^T$ and

$$A \ = \ \begin{bmatrix} K_2 - K_1 & 0 & 0 & \cdots & 0 \\ K_3 - K_1 & K_3 - K_2 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ K_n - K_1 & K_n - K_2 & K_n - K_3 & \cdots & 0 \\ 1 & 1 & 1 & \cdots & 1 \end{bmatrix}. \tag{3.10}$$

This formulation is obtained by removing the first two constraints of (3.8) which are redundant in this particular case.

Using this formulation and our earlier results, one can prove the following theorem giving necessary and sufficient conditions on a set of call option prices to prevent static arbitrage opportunities:

Theorem 3.3 *Let $K_1 < K_2 < \cdots < K_n$ denote the strike prices of European call options written on the same underlying security with the same maturity. There are no arbitrage opportunities if and only if the prices S_0^i satisfy the following conditions:*

1. $S_0^i > 0, \ i = 1, \ldots, n$
2. $S_0^i > S_0^{i+1}, \ i = 1, \ldots, n-1$
3. *The function $C(K_i) := S_0^i$ defined on the set $\{K_1, K_2, \ldots, K_n\}$ is a strictly convex function.*

3.3 Risk Measures: Conditional Value-at-Risk

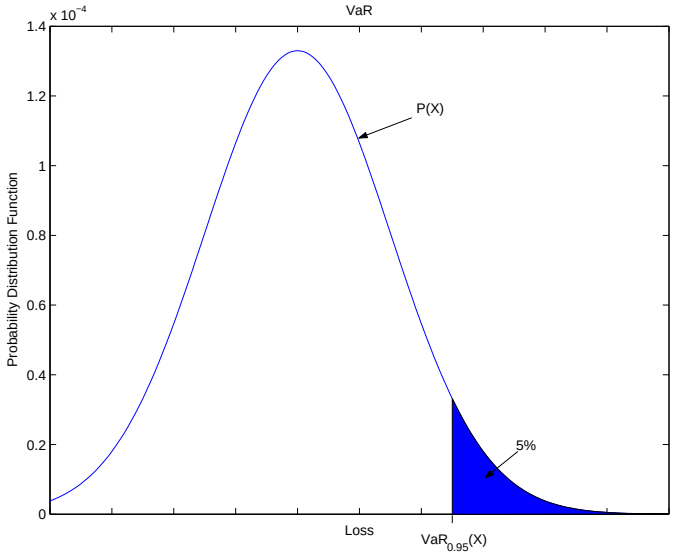
Financial activities involve risk. Our stock or mutual fund holdings carry the risk of losing value due to market conditions. Even money invested in a bank carries a risk–that of the bank going bankrupt and never returning the money let alone some interest. While individuals generally just have to live with such risks, financial and other institutions can and very often must manage risk using sophisticated mathematical techniques. Managing risk requires a good understanding of risk which comes from quantitative risk measures that adequately reflect the vulnerabilities of a company.

Perhaps the best-known risk measure is Value-at-Risk (VaR) developed by financial engineers at J.P. Morgan. VaR is a measure related to percentiles of loss distributions and represents the predicted maximum loss with a specified probability level (e.g., 95%) over a certain period of time (e.g., one day). Consider, for example, a random variable

X that represents loss from an investment portfolio over a fixed period of time. A negative value for X indicates gains. Given a probability level α, α-VaR of the random variable X is given by the following relation:

$$\text{VaR}_\alpha(X) \quad := \quad \min\{\gamma : P(X \leq \gamma) \geq \alpha\}. \tag{3.11}$$

The following figure illustrates the 0.95-VaR on a portfolio loss distribution plot:



VaR is widely used by people in the financial industry and VaR calculators are common features in most financial software. Despite this popularity, VaR has one important undesirable property–it lacks subadditivity. Risk measures should respect the maxim "diversification reduces risk" and therefore, satisfy the following property: "The total risk of two different investment portfolios does not exceed the sum of the individual risks." This is precisely what we mean by saying that a risk measure should be a subadditive function, i.e., for a risk measure f, we should have

$$f(x_1 + x_2) \leq f(x_1) + f(x_2), \forall x_1, x_2.$$

Consider the following simple example that illustrates that diversification can actually increase the risk measured by VaR:

Example 3.1 *Consider two independent investment opportunities each returning a \$1 gain with probability 0.96 and \$2 loss with probability 0.04. Then, 0.95-VaR for both investments are -1. Now consider the sum of these two investment opportunities. Because of independence, this sum has the following loss distribution: \$4 with probability* $0.04.04 = 0.0016$*, \$1 with probability* $2 \times 0.96 \times 0.04 = 0.0768$*, and -\$2 with probability* $0.96 \times 0.96 = 0.9216$*. Therefore, the 0.95-VaR of the sum of the two investments is 1, which exceeds -2, the sum of the 0.95-VaR values for individual investments.*

An additional difficulty with VaR is in its computation and optimization. When VaR is computed by generating scenarios, it turns out to be a non-smooth and non-convex function of the positions in the investment portfolio. Therefore, when one tries to optimize VaR computed in this manner, multiple local optimizers are encountered, hindering the global optimization process.

Another criticism on VaR is that it pays no attention to the magnitude of losses beyond the VaR value. This and other undesirable features of VaR led to the development of alternative risk measures. One well-known modification of VaR is obtained by computing the *expected* loss *given* that the loss exceeds VaR. This quantity is often called *conditional Value-at-Risk* or CVaR. There are several alternative names for this measure in the finance literature including Mean Expected Loss, Mean Shortfall, and Tail VaR. We now describe this risk measure in more detail and discuss how it can be optimized using linear programming techniques when the loss function is linear in the portfolio positions. Our discussion follows parts of articles by Rockafellar and Uryasev [12, 17].

We consider a portfolio of assets with random returns. We denote the portfolio choice vector with x and the random events by the vector y. Let $f(x, y)$ denote the loss function when we choose the portfolio x from a set X of feasible portfolios and y is the realization of the random events. We assume that the random vector y has a probability density function denoted by $p(y)$.

For a fixed decision vector x, we compute the cumulative distribution function of the loss associated with that vector x:

$$\Psi(x, \gamma) \;:=\; \int_{f(x,y)\leq\gamma} p(y)dy. \tag{3.12}$$

Then, for a given confidence level α, the α-VaR associated with portfolio x is given as

$$\mathrm{VaR}_\alpha(x) \;:=\; \min\{\gamma \in \Re : \Psi(x, \gamma) \geq \alpha\}. \tag{3.13}$$

Similarly, we define the α-CVaR associated with portfolio x:

$$\mathrm{CVaR}_\alpha(x) \;:=\; \frac{1}{1-\alpha}\int_{f(x,y)\geq \mathrm{VaR}_\alpha(x)} f(x, y)p(y)dy. \tag{3.14}$$

Note that,

$$\begin{aligned}
\mathrm{CVaR}_\alpha(x) &= \frac{1}{1-\alpha}\int_{f(x,y)\geq \mathrm{VaR}_\alpha(x)} f(x, y)p(y)dy \\
&\geq \frac{1}{1-\alpha}\int_{f(x,y)\geq \mathrm{VaR}_\alpha(x)} \mathrm{VaR}_\alpha(x)p(y)dy \\
&= \frac{\mathrm{VaR}_\alpha(x)}{1-\alpha}\int_{f(x,y)\geq \mathrm{VaR}_\alpha(x)} p(y)dy \\
&= \mathrm{VaR}_\alpha(x),
\end{aligned}$$

i.e., CVaR of a portfolio is always at least as big as its VaR. Consequently, portfolios with small CVaR also have small VaR. However, in general minimizing CVaR and VaR are not equivalent.

Since the definition of CVaR involves the VaR function explicitly, it is difficult to work with and optimize this function. Instead, we consider the following simpler auxiliary function:

$$F_\alpha(x,\gamma) \;:=\; \gamma + \frac{1}{1-\alpha}\int_{f(x,y)\geq\gamma} (f(x,y)-\gamma)\, p(y)dy. \tag{3.15}$$

Alternatively, we can write $F_{\alpha,x}(\gamma)$ as follows:

$$F_\alpha(x,\gamma) \;=\; \gamma + \frac{1}{1-\alpha}\int (f(x,y)-\gamma)^+ p(y)dy, \tag{3.16}$$

where $a^+ = \max\{a,0\}$. For a fixed $x \in X$, we also consider the following function of γ only:

$$F_{\alpha,x}(\gamma) \;:=\; F_\alpha(x,\gamma) = \gamma + \frac{1}{1-\alpha}\int_{f(x,y)\geq\gamma} (f(x,y)-\gamma)\, p(y)dy. \tag{3.17}$$

This final function of γ has the following important properties that makes it useful for the computation of VaR and CVaR:

1. $F_{\alpha,x}(\gamma)$ is a convex function of γ.
2. $\mathrm{VaR}_\alpha(x)$ is a minimizer of $F_{\alpha,x}(\gamma)$.
3. The minimum value of the function $F_{\alpha,x}(\gamma)$ is $\mathrm{CVaR}_\alpha(x)$.

As a consequence of the listed properties of the function $F_{\alpha,x}(\gamma)$ we immediately deduce that CVaR can be optimized via optimization of the function $F_\alpha(x,\gamma)$ with respect to x and γ simultaneously:

$$\min_{x\in X} \mathrm{CVaR}_\alpha(x) = \min_{x\in X}\min_{\gamma} F_{\alpha,x}(\gamma) = \min_{x\in X,\gamma} F_\alpha(x,\gamma). \tag{3.18}$$

Consequently, we can optimize CVaR directly, without needing to compute VaR first. If the loss function $f(x,y)$ is a convex (linear) function of the portfolio variables x, then $F_\alpha(x,\gamma)$ is also a convex (linear) function of x. In this case, provided the feasible portfolio set X is also convex, the optimization problems in (3.18) are smooth convex optimization problems that can be solved using well known optimization techniques for such problems.

Often it is not possible or desirable to compute/determine the joint density function $p(y)$ of the random events in our formulation. Instead, we may have a number of scenarios, say y_s for $s = 1,\ldots,S$, which may represent some historical values of the random events or some values obtained via computer simulation. In this case, we

obtain the following approximation to the function $F_\alpha(x,\gamma)$ by using the empirical distribution of the random events based on the available scenarios:

$$\tilde{F}_\alpha(x,\gamma) \;:=\; \gamma + \frac{1}{(1-\alpha)S}\sum_{s=1}^{S}(f(x,y_s)-\gamma)^+ . \tag{3.19}$$

Compare this definition to (3.16). Now, the problem $\min_{x\in X}\mathrm{CVaR}_\alpha(x)$ can be approximated by replacing $F_\alpha(x,\gamma)$ with $\tilde{F}_\alpha(x,\gamma)$ in (3.18):

$$\min_{x\in X,\gamma}\ \gamma + \frac{1}{(1-\alpha)S}\sum_{s=1}^{S}(f(x,y_s)-\gamma)^+ \tag{3.20}$$

To solve this optimization problem, we introduce artificial variables z_s to replace $(f(x,y_s)-\gamma)^+$. This is achieved by imposing the constraints $z_s \geq f(x,y_s)-\gamma$ and $z_s \geq 0$:

$$\begin{array}{lrcl} \min & \gamma + \frac{1}{(1-\alpha)S}\sum_{s=1}^{S} z_s & & \\ \text{s.t.} & z_s & \geq & 0,\ s=1,\ldots,S, \\ & z_s & \geq & f(x,y_s)-\gamma,\ s=1,\ldots,S, \\ & x \in X & & \end{array} \tag{3.21}$$

Note that the constraints $z_s \geq f(x,y_s)-\gamma$ and $z_s \geq 0$ alone can not ensure that $z_s = (f(x,y_s)-\gamma)^+ = \max\{f(x,y_s)-\gamma, 0\}$ since z_s can be larger than both right-hand-sides and be still feasible. However, since we are minimizing the objective function which involves a positive multiple of z_s, it will never be optimal to assign z_s a value larger than the maximum of the two quantities $f(x,y_s)-\gamma$ and 0, and therefore, in an optimal solution z_s will be precisely $(f(x,y_s)-\gamma)^+$, justifying our substitution.

In the case that $f(x,y)$ is linear in x, all the expressions $z_s \geq f(x,y_s)-\gamma$ represent linear constraints and therefore the problem (3.21) is a linear programming problem that can be solved using the simplex method or alternative LP algorithms.

Alternative optimization problems are often formulated within the context of risk management. For example, risk managers often try to optimize some performance measure (e.g., expected return) while making sure that certain risk measures do not exceed some threshold values. We may have the following problem when the risk measure is CVaR:

$$\begin{array}{lrcl} \max & \mu^T x & & \\ \text{s.t.} & \mathrm{CVaR}_{\alpha^j}(x) & \leq & U_{\alpha^j},\ j=1,\ldots,J. \end{array} \tag{3.22}$$

Above, J is an index set for different confidence levels used for CVaR computations and U_{α^j} represents the maximum tolerable CVaR value at the confidence level α^j. As above, we can replace the CVaR functions in the constraints of this problem with the function $F_\alpha(x,\gamma)$ as above and then approximate this function using the scenarios for random events. This approach results in the following approximation of the CVaR-constrained

problem (3.22):

$$\begin{array}{rrcl} \max & \mu^T x & & \\ \text{s.t.} & \gamma + \frac{1}{(1-\alpha^j)S}\sum_{s=1}^{S} z_s & \leq & U_{\alpha^j},\ j = 1, \ldots, J, \\ & z_s & \geq & 0,\ s = 1, \ldots, S, \\ & z_s & \geq & f(x, y_s) - \gamma,\ s = 1, \ldots, S, \\ & x \in X & & \end{array} \tag{3.23}$$

3.4 Exercises

1. Let S_0 be the current price of a security and assume that there are two possible prices for this security at the end of the current period: $S_1^u = S_0 \cdot u$ and $S_1^d = S_0 \cdot d$. (Assume $u > d$.) Also assume that there is a fixed interest paid on cash borrowed or lent at rate r for the given period. Let $R = 1 + r$. Show that if $u > R > d$ is not satisfied there is an arbitrage opportunity.

2. Recall the linear programming problem (3.9) that we developed to detect arbitrage opportunities in the prices European call options with a common underlying security and common maturity (but different strike prices). This formulation implicitly assumes that the i^{th} call can be bought or sold at the same current price of S_0^i. In real markets, there is always a gap between the price a buyer pays for a security and the amount the seller collects called the *bid-ask spread*. For a security purchase/sale to happen, a market intermediary collects purchase *bids* from potential buyers and *ask* prices from potential sellers. Then, the highest bid price is matched with the lowest ask price. Since all this happens in a transparent manner, bidders never bid more than the lowest asking price and the sellers never ask for less than the highest bid price. Therefore, the highest bid price is always less than the lowest ask price. The buyer pays the lowest ask price and the seller collects the highest bid price. The difference is collected by the marker intermediary that performs this transaction.

   Assume now that the ask price of the i^{th} call is given by S_a^i while its bid price is denoted by S_b^i with $S_a^i > S_b^i$. Develop an analogue of the LP (33) in the case where we can purchase the calls at their ask prices or sell them at their bid prices. Consider using two variables for each call option in your new LP.

3. Prove Theorem 3.3.

4. Consider all the call options on the S&P 500 index that are expiring n June 2003. Their current prices can be downloaded from the website of the Chicago Board of Options Exchange at `www.cboe.com` or several other market quote websites. Delayed data is available for free, so do not use the sites that charge fees for real-time data. Formulate the linear programming problem (33) (or, rather the

version you developed for Problem 4 since market quotes will include bid and ask prices) to determine whether these prices contain any arbitrage opportunities. Solve this linear programming problem using an LP software. Here are some suggestions:

- MATLAB has a linear programming solver that can be accessed with the command `linprog`. Type `help linprog` to find out details.
- If you do not have access to any linear programming software, you can use the website `http://www-neos.mcs.anl.gov/neos/` to access the Network Enable Optimization Server. Using this site, and their JAVA submission tool, you can submit a linear programming problem (in some standard format) and have a remote computer solve your problem using one of the several solver options. You will then receive the solution by e-mail.

Sometimes, illiquid securities (those that are not traded very often) can have misleading prices since the reported price corresponds to the last transaction in that security which may have happened several days ago, and if there were to be a new transaction, this value would change dramatically. As a result, it is quite possible that you will discover false "arbitrage opportunities" because of these misleading prices. Repeat the LP formulation and solve it again, this time only using prices of the call options that have had a trading volume of at least 100 on the day you downloaded the prices.

Chapter 4

Quadratic Programming: Theory and Algorithms

4.1 The Quadratic Programming Problem

As we discussed in the introductory chapter, quadratic programming (QP) refers to the problem of minimizing a quadratic function subject to linear equality and inequality constraints. In its standard form, this problem is represented as follows:

$$(\mathcal{QP}) \qquad \begin{array}{lrcl} \min_x & \frac{1}{2}x^TQx + c^Tx & & \\ & Ax & = & b \\ & x & \geq & 0, \end{array} \tag{4.1}$$

where $A \in I\!R^{m\times n}$, $b \in I\!R^m$, $c \in I\!R^n$, $Q \in I\!R^{n\times n}$ are given, and $x \in I\!R^n$.

Quadratic programming problems are encountered frequently in optimization models. For example, ordinary least squares problems are QPs with no constraints. Mean-variance optimization problems developed by Markowitz for the selection of *efficient* portfolios are QP problems. In addition, and perhaps more importantly, QP problems are solved as subproblems in the solution of general nonlinear optimization problems via *sequential quadratic programming* (SQP) approaches.

Recall that, when Q is a positive semidefinite matrix, i.e., when $y^TQy \geq 0$ for all y, the objective function of the problem $\mathcal{QP}$ is a convex function of x. Since the feasible set is a polyhedral set (i.e., a set defined by linear constraints) it is a convex set. Therefore, when Q is positive semidefinite, the QP (4.1) is a convex optimization problem. As such, its local optimal solutions are also global optimal solutions.

As in linear programming, we can develop a dual of quadratic programming problems. The dual of the problem (4.1) is given below:

$$(\mathcal{QD}) \quad \begin{array}{lrcccccl} \max_{x,y,s} & b^Ty & - & \frac{1}{2}x^TQx & & & & \\ & A^Ty & - & Qx & + & s & = & c \\ & & & x & , & s & \geq & 0. \end{array} \tag{4.2}$$

Note that, unlike the case of linear programming, the variables of the primal quadratic programming problem also appear in the dual QP.

4.2 Optimality Conditions

One of the fundamental tools in the study of optimization problems is the Karush-Kuhn-Tucker theorem that gives a list of conditions which are necessarily satisfied at any (local) optimal solution of a problem, provided that some mild regularity assumptions are satisfied. These conditions are commonly called KKT conditions and are provided in the Appendix.

Applying the KKT theorem to the QP problem (4.1), we obtain the following set of necessary conditions for optimality:

Theorem 4.1 *Suppose that x is a local optimal solution of the QP given in (4.1) so that it satisfies $Ax = b$, $x \geq 0$ and assume that Q is a positive semidefinite matrix. Then, there exists vectors y and s such that the following conditions hold:*

$$\begin{aligned} A^T y - Qx + s &= c && (4.3)\\ s &\geq 0 && (4.4)\\ x_i s_i &= 0, \forall i. && (4.5) \end{aligned}$$

Furthermore, x is a global optimal solution. In addition, if Q is positive definite, then x is uniquely determined.

Note that the positive definiteness condition related to the Hessian of the Lagrangian function in the KKT theorem is automatically satisfied for convex quadratic programming problems, and therefore is not included in Theorem 4.1. In the case that Q is positive definite, the objective function of (4.1) is strictly convex, and therefore, must have a unique minimizer.

Moreover, if vectors x, y, and s satisfy conditions (4.3)-(4.5) as well as primal feasibility conditions

$$\begin{aligned} Ax &= b && (4.6)\\ x &\geq 0 && (4.7) \end{aligned}$$

then, x is a global optimal solution of (4.1). In other words, conditions (4.3)-(4.7) are both necessary and sufficient for x, y, and s to describe a global optimal solution of the QP problem.

In a manner similar to linear programming, optimality conditions (4.3)-(4.7) can be seen as a collection of conditions for

1. primal feasibility: $Ax = b$, $x \geq 0$,
2. dual feasibility: $A^T y - Qx + s = c$, $s \geq 0$, and

3. complementary slackness: for each $i = 1, \ldots, n$ we have $x_i s_i = 0$.

Using this interpretation, one can develop modifications of the simplex method that can also solve convex quadratic programming problems. Here, we will describe an alternative algorithm that is based on Newton's method. Before describing the algorithm, let us write the optimality conditions in matrix form:

$$F(x, y, s) = \begin{bmatrix} A^T y - Qx + s - c \\ Ax - b \\ XSe \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, (x, s) \geq 0. \tag{4.8}$$

Above, X and S are diagonal matrices with the entries of the x and s vectors, respectively, on the diagonal, i.e., $X_{ii} = x_i$, and $X_{ij} = 0, i \neq j$, and similarly for s. Also, as before, e is an n-dimensional vector of ones.

4.3 Interior-Point Methods

We have seen that simplex method can be used to solve linear programming problems efficiently and its variants can be used for quadratic programming problems. Although these methods demonstrate satisfactory performance for the solution of most practical problems, they have the disadvantage that, in the worst case, the amount of time required to solve a given problem (the so-called *worst-case complexity*) can grow exponentially in the size of the problem. Here size refers to the number of variables and constraints in the problem.

One of the important concepts in the theoretical study of optimization algorithms is the concept of *polynomial-time algorithms*. This refers to an algorithm whose running time can be bounded by a polynomial function of the input size for all instances of the problem class that it is intended for.

After it was discovered in 1970s that the worst case complexity of the simplex method is exponential (and, therefore, that simplex method is not a polynomial-time algorithm) there was an effort to identify alternative methods for linear programming with polynomial time complexity. The first such method, called the *ellipsoid method* was developed by Yudin and Nemirovski in 1979. But the more exciting and enduring development was the announcement by Karmarkar in 1984 that an *interior-point method* can solve LPs in polynomial time. What distinguished Karmarkar's IPM from the ellipsoid method was that, in addition to having this desirable theoretical property, it could solve some LPs much faster than the simplex method.

The two decades that followed the publication of Karmarkar's paper has seen a very intense effort by the optimization research community to study theoretical and practical properties of IPMs. One of the early discoveries was that IPMs can be viewed as methods based on Newton's method but are modified to handle the inequality constraints. Some of the most important contributions were made by Nesterov and Nemirovski who showed that the IPM machinery can be applied to a much larger class

of problems than just LPs. Convex quadratic programming problems, for example, can be solved in polynomial time, as well as many other convex optimization problems using IPMs.

Here, we will describe a variant of IPMs for convex quadratic programming. Recall the optimality conditions of the QP problem in (4.1):

$$F(x,y,s) = \begin{bmatrix} A^Ty - Qx + s - c \\ Ax - b \\ XSe \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, (x,s) \geq 0. \tag{4.9}$$

The system of equations $F(x,y,s) = 0$ has $n+m+n$ variables and as many constraints. Because of the nonlinear equations $x_i s_i = 0$ we can not solve this system using Gaussian elimination type methods. But, since the system is square we can apply Newton's method. If there were no nonnegativity constraints, finding (x,y,s) satisfying these optimality conditions would be a straightforward exercise by applying Newton's method.

The existence of nonnegativity constraints creates a difficulty. In fact, the existence and the number of inequality constraints are among the most important factors that contribute to the difficulty of the solution of any optimization problem. Interior-point approach follows the following strategy to handle these inequality constraints: We first identify an initial solution (x^0, y^0, s^0) that satisfies the first two (linear) blocks of equations in $F(x,y,s) = 0$ (but not necessarily the third block $XSe = 0$), and also satisfies the nonnegativity constraints *strictly*, i.e., $x_0 > 0$ and $s_0 > 0$[1].

Once we find such an (x^0, y^0, s^0) we try to generate new points (x^k, y^k, s^k) that also satisfy these same conditions and get progressively closer to satisfying the third block of equations. This is achieved via careful application of a modified Newton's method.

Let us start by defining two sets related to the conditions (4.9):

$$\mathcal{F} \;:=\; \{(x,y,s) : Ax = b, A^Ty - Qx + s = c, x \geq 0, s \geq 0\} \tag{4.10}$$

is the set of *feasible points*, or simply the *feasible set*. Note that, we are using a primal-dual feasibility concept here. More precisely, since x variables come from the primal QP and (y,s) come from the dual QP, we impose both primal and dual feasibility conditions in the definition of $\mathcal{F}$. If $(x,y,s) \in \mathcal{F}$ also satisfy $x > 0$ and $s > 0$ we say that (x,y,s) is a *strictly feasible solution* and define

$$\mathcal{F}^o \;:=\; \{(x,y,s) : Ax = b, A^Ty - Qx + s = c, x > 0, s > 0\} \tag{4.11}$$

to be the *strictly feasible set*. In mathematical terms, $\mathcal{F}^o$ is the *relative interior* of the set $\mathcal{F}$.

[1]Notice that a point satisfying some inequality constraints strictly lies in the *interior* of the region defined by these inequalities–rather than being on the boundary. This is the reason why the method we are discussing is called an interior-point method.

IPMs we discuss here will generate iterates (x^k, y^k, s^k) that all lie in $\mathcal{F}^o$. Since we are generating iterates for both the primal and dual problems, this version of IPMs are often called *primal-dual interior-point methods*. Using this approach, we will obtain solutions for both the primal and dual problems at the end of the solution procedure. Solving the dual may appear to be a waste of time since we are only interested in the solution of the primal problem. However, years of computational experience demonstrated that primal-dual IPMs lead to the most efficient and robust implementations of the interior-point approach. This happens, because having some partial information on the dual problem (in the form of the dual iterates (y^k, s^k)) helps us make better and faster improvements on the iterates of the primal problem.

Iterative optimization algorithms have two essential components:

- a measure that can be used to evaluate the quality of alternative solutions and search directions
- a method to generate a better solution from a non-optimal solution.

As we stated before, IPMs rely on Newton's method to generate new estimates of the solutions. Let us discuss this more in depth. Ignore the inequality constraints in (4.9) for a moment, and focus on the nonlinear system of equations $F(x, y, s) = 0$. Assume that we have a current estimate (x^k, y^k, s^k) of the optimal solution to the problem. The Newton step from this point is determined by solving the following system of linear equations:

$$J(x^k, y^k, s^k) \begin{bmatrix} \Delta x^k \\ \Delta y^k \\ \Delta s^k \end{bmatrix} = -F(x^k, y^k, s^k), \tag{4.12}$$

where $J(x^k, y^k, s^k)$ is the Jacobian of the function F and $[\Delta x^k,\ \Delta y^k,\ \Delta s^k]^T$ is the search direction. First, observe that

$$J(x^k, y^k, s^k) = \begin{bmatrix} -Q & A^T & I \\ A & 0 & 0 \\ S^k & 0 & X^k \end{bmatrix} \tag{4.13}$$

where, X^k and S^k are diagonal matrices with the components of the vectors x^k and s^k along their diagonals. Furthermore, if $(x^k, y^k, s^k) \in \mathcal{F}^o$, then

$$F(x^k, y^k, s^k) = \begin{bmatrix} 0 \\ 0 \\ X^k S^k e \end{bmatrix} \tag{4.14}$$

and the Newton equation reduces to

$$\begin{bmatrix} -Q & A^T & I \\ A & 0 & 0 \\ S^k & 0 & X^k \end{bmatrix} \begin{bmatrix} \Delta x^k \\ \Delta y^k \\ \Delta s^k \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ -X^k S^k e \end{bmatrix}. \tag{4.15}$$

In standard Newton's method, once a Newton step is determined in this manner, one updates the current iterate with the Newton step to obtain the new iterate. In our case, this may not be permissible, since the Newton step may take us to a new point that does not necessarily satisfy the nonnegativity constraints $x \geq 0$ and $s \geq 0$. In our modification of Newton's method, we want to avoid such violations and therefore will seek a *step-size parameter* $\alpha_k \in (0, 1]$ such that $x^k + \alpha_k \Delta x^k > 0$ and $s^k + \alpha_k \Delta s^k > 0$. Note that the largest possible value of α_k satisfying these restrictions can be found using a procedure similar to the ratio test in simplex method. Once we determine the step-size parameter, we choose the next iterate as

$$(x^{k+1}, y^{k+1}, s^{k+1}) \;=\; (x^k, y^k, s^k) + \alpha_k(\Delta x^k, \Delta y^k, \Delta s^k).$$

If a value of α_k results in a next iterate $(x^{k+1}, y^{k+1}, s^{k+1})$ that is also in $\mathcal{F}^o$, we say that this value of α_k is *permissible*.

A naive modification of Newton's method as we described above is, unfortunately, not very good in practice since the permissible values of α_k are often too small and we can make very little progress toward the optimal solution. Therefore, one needs to modify the search direction as well as adjusting the step size along the direction. The usual Newton search direction obtained from (4.15) is called the *pure* Newton direction and we will consider *centered* Newton directions. To describe such directions, we first need to discuss the concept of the *central path*.

4.4 The Central Path

The central path $\mathcal{C}$ is a trajectory in the relative interior of the feasible region $\mathcal{F}^o$ that is very useful for both the theoretical study and also the implementation of IPMs. This trajectory is parametrized by a scalar $\tau > 0$, and the points (x_τ, y_τ, s_τ) on the central path are obtained as solutions of the following system:

$$F(x_\tau, y_\tau, s_\tau) = \begin{bmatrix} 0 \\ 0 \\ \tau e \end{bmatrix}, \; (x_\tau, s_\tau) > 0. \tag{4.16}$$

Then, the central path $\mathcal{C}$ is defined as

$$\mathcal{C} \;=\; \{(x_\tau, y_\tau, s_\tau) : \tau > 0\}. \tag{4.17}$$

The third block of equations in (4.16) can be rewritten as

$$(x_\tau)_i (s_\tau)_i = \tau, \; \forall i.$$

In other words, we no longer require that x and s are complementary vectors as in the optimality conditions, but we require the component products for these two vectors to

be equal for all components. Note that as $\tau \to 0$, the conditions (4.16) defining the points on the central path approximate the set of optimality conditions (4.9) more and more closely.

The system (4.16) has a unique solution for every $\tau > 0$, provided that $\mathcal{F}^o$ is nonempty. Furthermore, when $\mathcal{F}^o$ is nonempty, the trajectory (x_τ, y_τ, s_τ) converges to an optimal solution of the problem (4.1). The following figure depicts a sample feasible set and its central path.

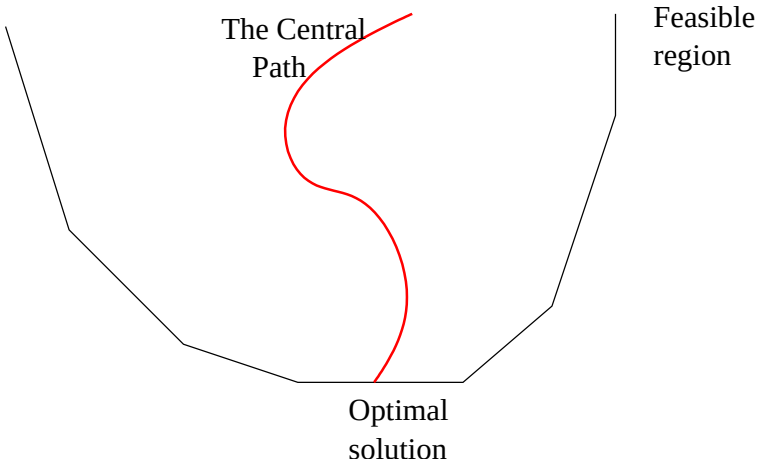


Figure 4.1: The Central Path

4.5 Interior-Point Methods

4.5.1 Path-Following Algorithms

As we mentioned above, when $\mathcal{F}^o$, the interior of the primal-dual feasible set is nonempty, the system (4.16) defining the central path has a unique solution for each positive τ. These solutions are called (primal-dual) central points and form the trajectory that we called the central path. Moreover, these solutions converge to optimal solutions of the primal-dual pair of quadratic programming problems. This observation suggests the following strategy for finding a solution of the system

$$F(x,y,s) = \begin{bmatrix} A^T y - Qx + s - c \\ Ax - b \\ XSe \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, (x,s) \geq 0. \tag{4.18}$$

that describes the optimality conditions for our QP: In an iterative manner, generate points that approximate central points for decreasing values of the parameter τ. Since

the central path converges to an optimal solution of the QP problem, these approximations to central points should also converge to a desired solution. This simple idea is the basis of interior-point *path-following algorithms* for optimization problems.

The strategy we outlined in the previous paragraph may appear confusing in a first reading. For example, one might wonder, why we would want to find approximations to central points, rather than central points themselves. Or, one might ask why we do not approximate or find the solutions of the optimality system (4.18) directly rather than generating all these intermediate iterates leading to such a solution. Let us respond to these potential questions. First of all, there is no good and computationally cheap way of solving (4.18) directly since it involves nonlinear equations of the form $x_i s_i = 0$. As we discussed above, if we apply Newton's method to the equations in (4.18), we run into trouble because of the additional nonnegativity constraints. In contrast, central points, being somewhat safely away from the boundaries defined by nonnegativity constraints, can be computed without most of the difficulties encountered in solving (4.18) directly. This is why we use central points for guidance. We are often satisfied with an approximation to a central point for reasons of computational efficiency. As the equations $(x_\tau)_i(s_\tau)_i = \tau$ indicate, central points are also defined by systems of nonlinear equations and additional nonnegativity conditions. Solving these systems exactly (or very accurately) can be as hard as solving the optimality system (4.18) and therefore would not be an acceptable alternative for a practical implementation. It is, however, relatively easy to find a well-defined approximation to central points (see the definition of the neighborhoods of the central path below), especially those that correspond to larger values of τ. Once we identify a point close to a central point on $\mathcal{C}$, we can do a clever and inexpensive search to find another point which is close to another central point on $\mathcal{C}$, corresponding to a smaller value of τ. Furthermore, this idea can be used repeatedly, resulting in approximations to central points with smaller and smaller τ values, allowing us to approach an optimal solution of the QP we are trying to solve. This is the essence of the path-following strategies.

4.5.2 Centered Newton directions

We will say that a Newton step used in an interior-point method is a *pure Newton step* if it is a step directed toward the optimal point satisfying $F(x, y, s) = [0,\ 0,\ 0]^T$. As we mentioned, these pure steps may be of poor quality in that they point toward the exterior of the feasible region. Instead, following the strategy we discussed in the previous paragraphs, most interior-point methods take a step toward points on the central path $\mathcal{C}$ corresponding to predetermined value of τ. Since such directions are aiming for central points, they are called *centered directions*. Figure 4.5.2 depicts a pure and centered Newton direction from a sample iterate.

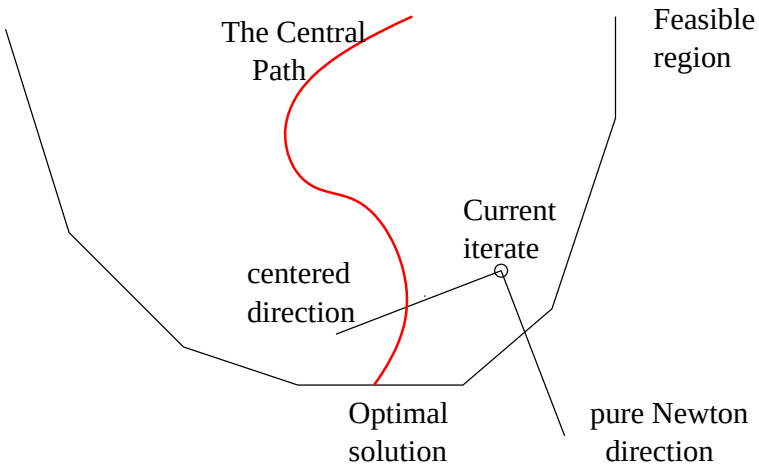


Figure 4.2: Pure and centered Newton directions

A centered direction is obtained by applying Newton update to the following system:

$$\hat{F}(x,y,s) = \begin{bmatrix} A^T y - Qx + s - c \\ Ax - b \\ XSe - \tau e \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}. \tag{4.19}$$

Since the Jacobian of $\hat{F}$ is identical to the Jacobian of F, proceeding as in equations (4.12)–(4.15), we obtain the following (modified) Newton equation for the centered direction:

$$\begin{bmatrix} -Q & A^T & I \\ A & 0 & 0 \\ S^k & 0 & X^k \end{bmatrix} \begin{bmatrix} \Delta x_c^k \\ \Delta y_c^k \\ \Delta s_c^k \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \tau e - X^k S^k e \end{bmatrix}. \tag{4.20}$$

We used the subscript c with the direction vectors to note that they are centered directions. Notice the similarity between (4.15) and (4.20).

One critical choice we need to make is the value of τ to be used in determining the centered direction. For this purpose, we first define the following measure often called the duality gap, or the average complementarity:

$$\mu = \mu(x,s) \;:=\; \frac{\sum_{i=1}^n x_i s_i}{n} = \frac{x^T s}{n}. \tag{4.21}$$

Note that, when (x,y,s) satisfy the conditions $Ax = b, x \geq 0$ and $A^T y - Qx + s = c, s \geq 0$, then (x,y,s) are optimal if and only if $\mu(x,s) = 0$. If μ is large, then we are far away from the solution. Therefore, μ serves as a measure of optimality for feasible points–the smaller the duality gap, the closer the point to optimality.

For a central point (x_τ, y_τ, s_τ) we have

$$\mu(x_\tau, s_\tau) \;=\; \frac{\sum_{i=1}^n (x_\tau)_i (s_\tau)_i}{n} = \frac{\sum_{i=1}^n \tau}{n} = \tau.$$

Because of this, we associate the central point (x_τ, y_τ, s_τ) with all feasible points (x, y, s) satisfying $\mu(x, s) = \tau$. All such points can be regarded as being at the same "level" as the central point (x_τ, y_τ, s_τ). When we choose a centered direction from a current iterate (x, y, s), we have the possibility of choosing to target a central point that is (i) at a lower level than our current point ($\tau < \mu(x, s)$), (ii) at the same level as our current point ($\tau = \mu(x, s)$), or (iii) at a higher level than our current point ($\tau > \mu(x, s)$). In most circumstances, the third option is not a good choice as it targets a central point that is "farther" than the current iterate to the optimal solution. Therefore, we will always choose $\tau \leq \mu(x, s)$ in defining centered directions. Using a simple change of variables, the centered direction can now be described as the solution of the following system:

$$\begin{bmatrix} -Q & A^T & I \\ A & 0 & 0 \\ S^k & 0 & X^k \end{bmatrix} \begin{bmatrix} \Delta x_c^k \\ \Delta y_c^k \\ \Delta s_c^k \end{bmatrix} \;=\; \begin{bmatrix} 0 \\ 0 \\ \sigma^k \mu^k e - X^k S^k e \end{bmatrix}, \tag{4.22}$$

where $\mu^k := \mu(x^k, s^k) = \frac{(x^k)^T s^k}{n}$ and $\sigma^k \in [0, 1]$ is a user defined quantity describing the ratio of the duality gap at the target central point and the current point.

When $\sigma^k = 1$ (equivalently, $\tau = \mu^k$ in our earlier notation), we have a *pure centering direction*. This direction does not intend to improve the duality gap and targets the central point whose duality gap is the same as our current iterate. Despite the lack of progress in terms of the duality gap, these steps are often desirable since large step sizes are permissible along such directions and points get well-centered so that the next iteration can make significant progress toward optimality. At the other extreme, we have $\sigma^k = 0$. This, as we discussed before, corresponds to the pure Newton step, also called the *affine-scaling direction*. Practical implementations often choose intermediate values for σ^k.

We are now ready to describe a generic interior-point algorithm that uses centered directions:

Algorithm 4.1 *Generic Interior Point Algorithm*

0. Choose $(x^0, y^0, s^0) \in \mathcal{F}^o$. For $k = 0, 1, 2, \ldots$ repeat the following steps.

1. Choose $\sigma^k \in [0, 1]$, let $\mu^k = \frac{(x^k)^T s^k}{n}$. Solve

$$\begin{bmatrix} -Q & A^T & I \\ A & 0 & 0 \\ S^k & 0 & X^k \end{bmatrix} \begin{bmatrix} \Delta x^k \\ \Delta y^k \\ \Delta s^k \end{bmatrix} \;=\; \begin{bmatrix} 0 \\ 0 \\ \sigma^k \mu^k e - X^k S^k e \end{bmatrix}.$$

2. *Choose* α^k *such that*

$$x^k + \alpha^k \Delta x^k > 0, \quad \textit{and } s^k + \alpha^k \Delta s^k > 0.$$

Set

$$(x^{k+1}, y^{k+1}, s^{k+1}) \;=\; (x^k, y^k, s^k) + \alpha_k(\Delta x^k, \Delta y^k, \Delta s^k),$$

and $k = k + 1$.

4.5.3 Neighborhoods of the Central Path

Variants of interior-point methods differ in the way they choose the centering parameter σ^k and the step-size parameter α^k in each iteration. Path-following methods, as we have been discussing, aim to generate iterates that are approximations to the central points. This is achieved by a careful selection of the centering and step-size parameters. Before we discuss the selection of these parameters let us make the notion of "approximate central points" more precise.

Recall that central points are those in the set $\mathcal{F}^o$ that satisfy the additional conditions that $x_i s_i = \tau, \forall i$, for some positive τ. Consider a central point (x_τ, y_τ, s_τ). If a point (x, y, s) approximates this central point, we would expect that the Euclidean distance between these two points is small, i.e.,

$$\|(x, y, s) - (x_\tau, y_\tau, s_\tau)\|$$

is small. Then, the set of approximations to (x_τ, y_τ, s_τ) may be defined as:

$$\{(x, y, s) \in \mathcal{F}^o : \|(x, y, s) - (x_\tau, y_\tau, s_\tau)\| \leq \varepsilon\}, \tag{4.23}$$

for some $\varepsilon \geq 0$. Note, however, that it is difficult to obtain central points explicitly. Instead, we have their implicit description through the system (4.19). Therefore, a description such as (4.23) is of little practical/algorithmic value when we do not know (x_τ, y_τ, s_τ). Instead, we consider descriptions of sets that *imply* proximity to central points. Such descriptions are often called the *neighborhoods* of the central path. Two of the most commonly used neighborhoods of the central path are:

$$\mathcal{N}_2(\theta) \;:=\; \{(x, y, s) \in \mathcal{F}^o : \|XSe - \mu e\| \leq \theta\mu,\ \mu = \frac{x^T s}{n}\}, \tag{4.24}$$

for some $\theta \in (0, 1)$ and

$$\mathcal{N}_{-\infty}(\gamma) \;:=\; \{(x, y, s) \in \mathcal{F}^o : x_i s_i \geq \gamma\mu\ \forall i,\ \mu = \frac{x^T s}{n}\}, \tag{4.25}$$

for some $\gamma \in (0, 1)$. The first neighborhood is called the 2-norm neighborhood while the second one the one-sided ∞-norm neighborhood (but often called the $-\infty$-norm

neighborhood, hence the notation). One can guarantee that the generated iterates are "close" to the central path by making sure that they all lie in one of these neighborhoods. Note that if we choose $\theta = 0$ in (4.24) or $\gamma = 1$ in (4.25), the neighborhoods we defined degenerate to the central path $\mathcal{C}$.

For typical values of θ and γ, the 2-norm neighborhood is often much smaller than the $-\infty$-norm neighborhood. Indeed,

$$\|XSe - \mu e\| \leq \theta\mu \Leftrightarrow \left\| \begin{array}{c} \frac{x_1 s_1}{\mu} - 1 \\ \frac{x_2 s_2}{\mu} - 1 \\ \vdots \\ \frac{x_n s_n}{\mu} - 1 \end{array} \right\| \leq \theta, \tag{4.26}$$

which, in turn, is equivalent to

$$\sum_{i=1}^{n} \left(\frac{x_i s_i}{\mu} - 1 \right)^2 \leq \theta^2.$$

In this last expression, the quantity $\frac{x_i s_i}{\mu} - 1 = \frac{x_i s_i - \mu}{\mu}$ is the relative deviation of $x_i s_i$'s from their average value μ. Therefore, a point is in the 2-norm neighborhood only if the sum of the squared relative deviations is small. Thus, $\mathcal{N}_2(\theta)$ contains only a small fraction of the feasible points, even when θ is close to 1. On the other hand, for the $-\infty$-norm neighborhood, the only requirement is that each $x_i s_i$ should not be much smaller than their average value μ. For small (but positive) γ, $\mathcal{N}_{-\infty}(\gamma)$ may contain almost the entire set $\mathcal{F}^o$.

In summary, 2-norm neighborhoods are narrow while the $-\infty$-norm neighborhoods are relatively wide. The practical consequence of this observation is that, when we restrict our iterates to be in the 2-norm neighborhood of the central path as opposed to the $-\infty$-norm neighborhood, we have much less room to maneuver and our step-sizes may be cut short. The next figure illustrates this behavior. For these reasons, algorithms using the narrow 2-norm neighborhoods are often called *short-step path-following methods* while the methods using the wide $-\infty$-norm neighborhoods are called *long-step path-following methods*

The price we pay for the additional flexibility with wide neighborhoods come in the theoretical worst-case analysis of algorithms using such neighborhoods. When the iterates are restricted to the 2-norm neighborhood, we have a stronger control of the iterates as they are very close to the central path– a trajectory with many desirable theoretical features. Consequently, we can guarantee that even in the worst case the iterates that lie in the 2-norm neighborhood will converge to an optimal solution relatively fast. In contrast, iterates that are only restricted to a $-\infty$-norm neighborhood can get relatively far away from the central path and may not possess its nice theoretical properties. As a result, iterates may "get stuck" in undesirable corners of the feasible set and the convergence may be slow in these worst-case scenarios. Of course, the worst case scenarios rarely happen and typically (on average) we see faster convergence with long-step methods than with short-step methods.

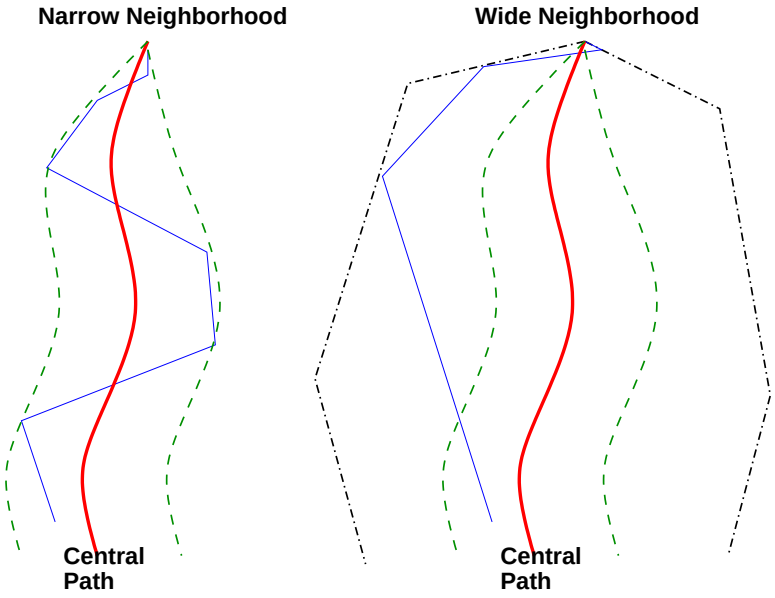


Figure 4.3: Narrow and wide neighborhoods of the central path

4.5.4 A Long-Step Path-Following Algorithm

Next, we formally describe a long-step path following algorithm that specifies some of the parameter choices of the generic algorithm we described above.

Algorithm 4.2 *Long-Step Path-Following Algorithm*

0. Given $\gamma \in (0,1), 0 < \sigma_{\min} < \sigma_{\max} < 1$, *choose* $(x^0, y^0, s^0) \in \mathcal{N}_{-\infty}(\gamma)$. *For* $k = 0, 1, 2, \ldots$ *repeat the following steps.*

1. Choose $\sigma^k \in [\sigma_{\min}, \sigma_{\max}]$, *let* $\mu^k = \frac{(x^k)^T s^k}{n}$. *Solve*

$$\begin{bmatrix} -Q & A^T & I \\ A & 0 & 0 \\ S^k & 0 & X^k \end{bmatrix} \begin{bmatrix} \Delta x^k \\ \Delta y^k \\ \Delta s^k \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \sigma^k \mu^k e - X^k S^k e \end{bmatrix}.$$

2. Choose α^k *such that*

$$(x^k, y^k, s^k) + \alpha_k(\Delta x^k, \Delta y^k, \Delta s^k) \in \mathcal{N}_{-\infty}(\gamma).$$

Set

$$(x^{k+1}, y^{k+1}, s^{k+1}) = (x^k, y^k, s^k) + \alpha_k(\Delta x^k, \Delta y^k, \Delta s^k),$$

and $k = k + 1$.

4.5.5 Starting from an Infeasible Point

Both the generic interior-point method and the long-step path-following algorithm we described above require that one starts with a strictly feasible iterate. This requirement is not practical since finding such a starting point is not always a trivial task. Fortunately, however, we can accommodate infeasible starting points with a small modification of the linear system we solve in each iteration.

For this purpose, we only require that the initial point (x^0, y^0, s^0) satisfy the nonnegativity restrictions strictly: $x^0 > 0$ and $s^0 > 0$. Such points can be generated trivially. We are still interested in solving the following nonlinear system:

$$\hat{F}(x, y, s) = \begin{bmatrix} A^T y - Qx + s - c \\ Ax - b \\ XSe - \tau e \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \tag{4.27}$$

as well as $x \geq 0$, $s \geq 0$. As in (59), the Newton step from an infeasible point (x^k, y^k, s^k) is determined by solving the following system of linear equations:

$$J(x^k, y^k, s^k) \begin{bmatrix} \Delta x^k \\ \Delta y^k \\ \Delta s^k \end{bmatrix} = -\hat{F}(x^k, y^k, s^k), \tag{4.28}$$

which reduces to

$$\begin{bmatrix} -Q & A^T & I \\ A & 0 & 0 \\ S^k & 0 & X^k \end{bmatrix} \begin{bmatrix} \Delta x^k \\ \Delta y^k \\ \Delta s^k \end{bmatrix} = \begin{bmatrix} c + Qx^k - A^T y^k - s^k \\ b - Ax^k \\ \tau e - X^k S^k e \end{bmatrix}. \tag{4.29}$$

We no longer have zeros in the first and second blocks of the right-hand-side vector since we are not assuming that the iterates satisfy $Ax^k = b$ and $A^T y^k - Qx^k + s^k = c$. Replacing the linear system in the two algorithm descriptions above with (4.29) we obtain versions of these algorithms that work with infeasible iterates. In these versions of the algorithms, search for feasibility and optimality are performed simultaneously.

4.6 QP software

As for linear programs, there are several software options for solving practical quadratic programming problems. Many of the commercial software options are very efficient and solve very large QPs within seconds or minutes. A somewhat dated survey of nonlinear programming software, which includes software designed for QPs, can be found at

`http://www.lionhrtpub.com/orms/surveys/nlp/nlp.html`.

The "Optimization Software Guide" website we mentioned when we discussed LP software is also useful for QP solvers. You can reach this guide at

`http://www-fp.mcs.anl.gov/otc/Guide/SoftwareGuide/index.html`.

LOQO is a very efficient and robust interior-point based software for QPs and other nonlinear programming problems. It is available from

`http://www.orfe.princeton.edu/~loqo/`.

OOQP is an object-oriented C++ package, based on a primal-dual interior-point method, for solving convex quadratic programming problems (QPs). It contains code that can be used "out of the box" to solve a variety of structured QPs, including general sparse QPs, QPs arising from support vector machines, Huber regression problems, and QPs with bound constraints. It is available for free from the following website:

`http://www.cs.wisc.edu/~swright/ooqp/`

4.7 Exercises

1. In the study of interior-point methods for solving quadratic programming problems we encountered the following matrix:

$$M := \begin{bmatrix} -Q & A^T & I \\ A & 0 & 0 \\ S^k & 0 & X^k \end{bmatrix},$$

where (x^k, y^k, s^k) is the current iterate, X^k and S^k are diagonal matrices with the components of the vectors x^k and s^k along their diagonals. Recall that M is the Jacobian matrix of the function that defines the optimality conditions of the QP problem. This matrix appears in linear systems we need to solve in each interior-point iteration. We can solve these systems only when M is nonsingular. Show that M is necessarily nonsingular when A has full row rank and Q is positive semidefinite. Provide an example with a Q matrix that is not positive semidefinite (but A matrix has full row rank) such that M is singular. (Hint: To prove non-singularity of M when Q is positive semidefinite and A has full row rank, consider a solution of the system

$$\begin{bmatrix} -Q & A^T & I \\ A & 0 & 0 \\ S^k & 0 & X^k \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta s \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}.$$

It is sufficient to show that the only solution to this system is $\Delta x = 0, \Delta y = 0, \Delta s = 0$. To prove this, first eliminate Δs variables from the system, and then eliminate Δx variables.)

2. When we discussed path-following methods for quadratic programming problems, we talked about the central path and the following two (classes of) neighborhoods of the central path:

$$\mathcal{N}_2(\theta) \;:=\; \{(x, y, s) \in \mathcal{F}^o : \|XSe - \mu e\| \leq \theta\mu,\ \mu = \frac{x^T s}{n}\},$$

for some $\theta \in (0,1)$ and

$$\mathcal{N}_{-\infty}(\gamma) \;:=\; \{(x,y,s) \in \mathcal{F}^o : x_i s_i \geq \gamma\mu \;\forall i,\; \mu = \frac{x^T s}{n}\},$$

for some $\gamma \in (0,1)$.

(i) Show that $\mathcal{N}_2(\theta_1) \subset \mathcal{N}_2(\theta_2)$ when $0 < \theta_1 \leq \theta_2 < 1$, and that $\mathcal{N}_{-\infty}(\gamma_1) \subset \mathcal{N}_{-\infty}(\gamma_2)$ for $0 < \gamma_2 \leq \gamma_1 < 1$.

(ii) Show that $\mathcal{N}_2(\theta) \subset \mathcal{N}_{-\infty}(\gamma)$ if $\gamma \leq 1 - \theta$.

3. Consider the following quadratic programming formulation obtained from a small portfolio selection model:

$$\begin{aligned} \min_x \, & [x_1 \; x_2 \; x_3 \; x_4] \begin{bmatrix} 0.01 & 0.005 & 0 & 0 \\ 0.005 & 0.01 & 0 & 0 \\ 0 & 0 & 0.04 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} \\ & x_1 + x_2 + x_3 = 1 \\ & -x_2 + x_3 + x_4 = 0.1 \\ & x_1, x_2, x_3, x_4 \geq 0. \end{aligned}$$

We have the following iterate for this problem:

$$x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 1/3 \\ 1/3 \\ 1/3 \\ 0.1 \end{bmatrix}, \quad y = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} 0.001 \\ -0.001 \end{bmatrix}, \quad s = \begin{bmatrix} s_1 \\ s_2 \\ s_3 \\ s_4 \end{bmatrix} = \begin{bmatrix} 0.004 \\ 0.003 \\ 0.0133 \\ 0.001 \end{bmatrix}.$$

Verify that $(x,y,s) \in \mathcal{F}^o$. Is this point on the central path? Is it on $\mathcal{N}_{-\infty}(0.1)$? How about $\mathcal{N}_{-\infty}(0.05)$? Compute the pure centering ($\sigma = 1$) and pure Newton ($\sigma = 0$) directions from this point. For each direction, find the largest step-size α that can be taken along that direction without leaving the neighborhood $\mathcal{N}_{-\infty}(0.05)$? Comment on your results.

Chapter 5

QP Models and Tools in Finance

5.1 Mean-Variance Optimization

In the introductory chapter, we have discussed Markowitz' theory of mean-variance optimization (MVO) for the selection of portfolios of securities (or asset classes) in a manner that trades off the expected returns and the perceived risk of potential portfolios.

Consider assets $S_1, S_2, \ldots, S_n$ ($n \geq 2$) with random returns. Let μ_i and σ_i denote the expected return and the standard deviation of the return of asset S_i. For $i \neq j$, ρ_{ij} denotes the correlation coefficient of the returns of assets S_i and S_j. Let $\mu = [\mu_1, \ldots, \mu_n]^T$, and Q be the $n \times n$ symmetric covariance matrix with $Q_{ii} = \sigma_i^2$ and $Q_{ij} = \rho_{ij}\sigma_i\sigma_j$ for $i \neq j$. Denoting the proportion of the total funds invested in security i by x_i, one can represent the expected return and the variance of the resulting portfolio $x = (x_1, \ldots, x_n)$ as follows:

$$E[x] = x_1\mu_1 + \ldots + x_n\mu_n = \mu^T x,$$

and

$$Var[x] = \sum_{i,j} \rho_{ij}\sigma_i\sigma_j x_i x_j = x^T Q x,$$

where $\rho_{ii} \equiv 1$.

Since variance is always nonnegative, it follows that $x^T Q x \geq 0$ for any x, i.e., Q is positive semidefinite. We will assume that it is in fact positive definite, which is essentially equivalent to assuming that there are no redundant assets in our collection $S_1, S_2, \ldots, S_n$. We further assume that the set of *admissible* portfolios is a nonempty polyhedral set and represent it as $\mathcal{X} := \{x : Ax = b, Cx \geq d\}$, where A is an $m \times n$ matrix, b is an m-dimensional vector, C is a $p \times n$ matrix and d is a p-dimensional vector. This representation lets us treat any linear portfolio constraint such as short-sale restrictions or limits on asset/sector allocations in a unified manner.

Recall that a feasible portfolio x is called *efficient* if it has the maximal expected return among all portfolios with the same variance, or alternatively, if it has the min-

imum variance among all portfolios that have at least a certain expected return. The collection of efficient portfolios form the *efficient frontier* of the portfolio universe. The efficient frontier is often represented as a curve in a two-dimensional graph where the coordinates of a plotted point corresponds to the expected return and the standard deviation on the return of an efficient portfolio.

Since we assume that Q is positive definite, the variance is a strictly convex function of the portfolio variables and there exists a *unique* portfolio in $\mathcal{X}$ that has the minimum variance. Let us denote this portfolio with $x_{\min}$ and its return $\mu^T x_{\min}$ with $R_{\min}$. Note that $x_{\min}$ is an efficient portfolio. We let $R_{\max}$ denote the maximum return for an admissible portfolio which can be $+\infty$.

Markowitz' *mean-variance optimization* (MVO) problem can be formulated in three different but equivalent ways. We have seen one of these formulations in the first chapter: Find the minimum variance portfolio of the securities 1 to n that yields at least a target value of expected return (say b). Mathematically, this formulation produces a quadratic programming problem:

$$\begin{array}{rrcl} \min_x & \frac{1}{2}x^T Q x & & \\ & \mu^T x & \geq & R \\ & Ax & = & b \\ & Cx & \geq & d. \end{array} \tag{5.1}$$

The first constraint indicates that the expected return is no less than the target value R. Solving this problem for values of R ranging between $R_{\min}$ and $R_{\max}$ one obtains all efficient portfolios. As we discussed above, the objective function corresponds to (one half) the total variance of the portfolio. The constant $\frac{1}{2}$ is added for convenience in the optimality conditions–it obviously does not affect the optimal solution.

This is a convex quadratic programming problem for which the first order conditions are both necessary and sufficient for optimality. We present these conditions next. x_R is an optimal solution of problem (5.1) if and only if there exists $\lambda_R \in \Re$, $\gamma_E \in \Re^m$, and $\gamma_I \in \Re^p$ satisfying the following conditions:

$$\begin{array}{c} Qx_R - \lambda_R \mu - A^T \gamma_E - C^T \gamma_I = 0, \\ \mu^T x_R \geq R, \quad Ax_R = b, \quad Cx_R \geq d, \\ \lambda_R \geq 0, \quad \lambda_R(\mu^T x_R - R) = 0, \\ \gamma_I \geq 0, \quad \gamma_I^T (Cx_R - d) = 0. \end{array} \tag{5.2}$$

5.2 Maximizing the Sharpe Ratio

Consider the setting in the previous subsection. Let us define the function $\sigma(R) : [R_{\min}, R_{\max}] \to \Re$ as $\sigma(R) := (x_R^T Q x_R)^{1/2}$, where x_R denotes the unique solution of problem (5.1). Since we assumed that Q is positive definite, it is easily shown that the function $\sigma(R)$ is strictly convex in its domain. As mentioned before, the efficient frontier is the graph $E = \{(R, \sigma(R)) : R \in [R_{\min}, R_{\max}]\}$.

We now consider a *riskless asset* whose expected return is $r_f \geq 0$. We will assume that $r_f < R_{\min}$, which is natural since the portfolio $x_{\min}$ has a positive risk associated with it while the riskless asset does not.

Return/risk profiles of different combinations of a risky portfolio with the riskless asset can be represented as a straight line—a *capital allocation line (CAL)*—on the mean vs. standard deviation graph. The optimal CAL is the CAL that lies below all the other CALs for $R > r_f$ since the corresponding portfolios will have the lowest standard deviation for any given value of $R > r_f$. Then, it follows that this optimal CAL goes through a point on the efficient frontier and never goes above a point on the efficient frontier. In other words, the slope of the optimal CAL is a *sub-derivative* of the function $\sigma(R)$ that defines the efficient frontier. The point where the optimal CAL touches the efficient frontier corresponds to the optimal risky portfolio.

Alternatively, one can think of the optimal CAL as the CAL with the smallest slope. Mathematically, this can be expressed as the portfolio x that maximizes the quantity

$$h(x) = \frac{\mu^T x - r_f}{(x^T Q x)^{1/2}},$$

among all $x \in S$. This quantity is precisely the *reward-to-variability ratio* introduced by Sharpe to measure the performance of mutual funds [15]. Now more commonly known as the Sharpe measure, or Sharpe ratio, this quantity measures the expected return per unit of risk (standard deviation) for a zero-investment strategy. The portfolio that maximizes the Sharpe ratio is found by solving the following problem:

$$\begin{array}{lrcl} \max_x & \frac{\mu^T x - r_f}{(x^T Q x)^{1/2}} & & \\ & Ax & = & b \\ & Cx & \geq & d. \end{array} \tag{5.3}$$

In this form, this problem is not easy to solve. Although it has a nice, polyhedral feasible region, its objective function is somewhat complicated, and worse, is possibly non-concave. Therefore, (5.3) is not a convex optimization problem. The standard strategy to find the portfolio maximizing the Sharpe ratio, often called the *optimal risky portfolio*, is the following: First, one traces out the efficient frontier on a two dimensional return vs. standard deviation graph. Then, the point on this graph corresponding to the optimal risky portfolio is found as the tangency point of the line going through the point representing the riskless asset and is tangent to the efficient frontier. Once this point is identified, one can recover the composition of this portfolio from the information generated and recorded while constructing the efficient frontier.

Here, we describe a direct method to obtain the optimal risky portfolio by constructing a convex quadratic programming problem equivalent to (5.3). The only assumption we need is that $\sum_{i=1}^n x_i = 1$ for any feasible portfolio x. This is a natural assumption since x_i's correspond to the proportions of the portfolio in different securities/asset classes.

First, observe that using the relation $e^T x = 1$ with $e = [1\ 1\ \ldots\ 1]^T$, $h(x)$ can be rewritten as a homogeneous function of x–we call this function $g(x)$:

$$h(x) = \frac{\mu^T x - r_f}{\sqrt{x^T Q x}} = \frac{(\mu - r_f e)^T x}{\sqrt{x^T Q x}} =: g(x) = g(\frac{x}{\kappa}), \quad \forall \kappa > 0.$$

The vector $\mu - r_f e$ is the vector of returns in excess of the risk-free lending rate.

Next, we homogenize $\mathcal{X} = \{x : Ax = b, Cx \geq d\}$ applying the *lifting* technique to it, i.e., we consider a set $\mathcal{X}^+$ that lives in a one higher dimensional space than $\mathcal{X}$ and is defined as follows:

$$\mathcal{X}^+ \ := \ \{x \in \Re^n, \kappa \in \Re | \kappa > 0, \frac{x}{\kappa} \in \mathcal{X}\} \cup (0,0). \tag{5.4}$$

We add the vector $(0,0)$ to the set to achieve a closed set. Note that $\mathcal{X}^+$ is a cone. For example, when $\mathcal{X}$ is a circle, $\mathcal{X}^+$ resembles an ice-cream cone. When $\mathcal{X}$ is polyhedral, e.g., $\mathcal{X} = \{x | Ax \geq b, Cx = d\}$, we have $\mathcal{X}^+ = \{(x,\kappa) | Ax - b\kappa \geq 0, Cx - d\kappa = 0, \kappa \geq 0\}$. Now, using the observation that $h(x) = g(x), \forall x \in \mathcal{X}$ and that $g(x)$ is homogeneous, we conclude that (5.3) is equivalent to

$$\max g(x) \text{ s.t. } (x,\kappa) \in \mathcal{X}^+. \tag{5.5}$$

Again, using the observation that $g(x)$ is homogeneous in x, we see that adding the normalizing constraint $(\mu - r_f e)^T x = 1$ to (5.5) does not affect the optimal solution–from among a ray of optimal solutions, we will find the one on the normalizing hyperplane. Note that for any $x \in \mathcal{X}$ with $(\mu - r_f e)^T x > 0$, the normalizing hyperplane will intersect with an $(x^+, \kappa^+) \in \mathcal{X}^+$ such that $x = x^+/\kappa^+$–in fact, $x^+ = \frac{x}{(\mu - r_f e)^T x}$ and $\kappa^+ = \frac{1}{(\mu - r_f e)^T x}$. The normalizing hyperplane will miss the rays corresponding to points in $\mathcal{X}$ with $(\mu - r_f e)^T x \leq 0$, but since they can not be optimal, this will not affect the optimal solution. Therefore, substituting $(\mu - r_f e)^T x = 1$ into $g(x)$ we obtain the following equivalent problem:

$$\max \frac{1}{\sqrt{x^T Q x}} \text{ s.t. } (x,\kappa) \in \mathcal{X}^+, \ (\mu - r_f e)^T x = 1. \tag{5.6}$$

Thus, we proved the following result:

Proposition 5.1 *Given a set $\mathcal{X}$ of feasible portfolios with the property that $e^T x = 1$, $\forall x \in \mathcal{X}$, the portfolio x^* with the maximum Sharpe ratio in this set can be found by solving the following problem with a convex quadratic objective function*

$$\min x^T Q x \ s.t. \ (x,\kappa) \in \mathcal{X}^+, \ (\mu - r_f e)^T x = 1, \tag{5.7}$$

with $\mathcal{X}^+$ as in (5.4). If $(\hat{x}, \hat{\kappa})$ is the solution to (5.7), then $x^* = \frac{\hat{x}}{\hat{\kappa}}$. □

This last problem can be solved using the techniques we discussed for convex quadratic programming problems.

5.3 Returns-Based Style Analysis

In two ground-breaking articles, Sharpe described how constrained optimization techniques can be used to determine the effective asset mix of a fund using only the return time series for the fund and a number of carefully chosen asset classes [13, 14]. Often, passive indices or index funds are used to represent the chosen asset classes and one tries to determine a portfolio of these funds/indices whose returns provide the best match for the returns of the fund being analyzed. The allocations in the portfolio can be interpreted as the fund's style and consequently, this approach has become to known as *returns-based style analysis*, or RBSA.

RBSA provides an inexpensive and timely alternative to *fundamental analysis* of a fund to determine its style/asset mix. Fundamental analysis uses the information on actual holdings of a fund to determine its asset mix. When all the holdings are known, the asset mix of the fund can be inferred easily. However, this information is rarely available, and when it is available, it is often quite expensive and/or several weeks or months old. Since RBSA relies only on returns data which is immediately available, and well-known optimization techniques, it can be employed in circumstances where fundamental analysis cannot be used.

The mathematical model for RBSA is surprisingly simple. It uses the following generic linear factor model: Let R_t denote the return of a security–usually a mutual fund, but can be an index, etc.–in period t for $t = 1, \ldots, T$ where T corresponds to the number of periods in the modeling window. Further, let F_{it} denote the return on factor i in period t, for $i = 1, \ldots, n$, $t = 1, \ldots, T$. Then, R_t can be represented as follows:

$$\begin{aligned} R_t &= w_{1t}F_{1t} + w_{2t}F_{2t} + \ldots + w_{nt}F_{nt} + \epsilon_t \\ &= F_t w_t + \epsilon_t, \ t = 1, \ldots, T. \end{aligned} \tag{5.8}$$

In this equation, w_{it} quantities represent the sensitivities of R_t to each one of the n factors, and ϵ_t represents the non-factor return. We use the notation $w_t = \left[\ w_{1t}, \ldots, w_{nt} \ \right]^T$ and $F_t = \left[\ F_{1t}, \ldots, F_{nt} \ \right]$.

The linear factor model (5.8) has the following convenient interpretation when the factor returns F_{it} correspond to the returns of passive investments, such as those in an index fund for an asset class: One can form a benchmark portfolio of the passive investments (with weights w_{it}) and the difference between the fund return R_t and the return of the benchmark portfolio $F_t w_t$ is the non-factor return contributed by the fund manager using stock selection, market timing, etc. In other words, ϵ_t represents the additional return resulting from active management of the fund. Of course, this additional return can be negative.

The benchmark portfolio return interpretation for the quantity $F_t w_t$ suggests that one should choose the sensitivities (or weights) w_{it} such that they are all nonnegative and sum to zero. With these constraints in mind, Sharpe proposes to choose w_{it} to minimize the variance of the non-factor return ϵ_t. In his model, Sharpe restricts the

weights to be constant over the period in consideration so that w_{it} does not depend on t. In this case, we use $w = \left[\ w_1, \ldots, w_n\ \right]^T$ to denote the time-invariant factor weights and formulate the following quadratic programming problem:

$$\begin{array}{lll} \min_{w\in\Re^n} & \mathrm{var}(\epsilon_t) = \mathrm{var}(R_t - F_t w) & \\ \text{s.t.} & \sum_{i=1}^n w_i &= 1 \\ & w_i &\geq 0, \forall i. \end{array} \tag{5.9}$$

The objective of minimizing the variance of the non-factor return ϵ_t deserves some comment. Since we are essentially formulating a tracking problem, and since ϵ_t represents the "tracking error", one may be tempted to minimize the magnitude of this quantity rather than its variance. Since the Sharpe model interprets the quantity ϵ_t as a consistent management effect, the objective is to determine a benchmark portfolio such that the difference between fund returns and the benchmark returns is as close to constant (i.e., variance 0) as possible. So, we want the fund return and benchmark return graphs to show two almost parallel lines with the distance corresponding to manager's consistent contribution to the fund return. This objective is almost equivalent to choosing weights in order to maximize the R^2 of this regression model. The equivalence is not exact since we are using constrained regression and this may lead to correlation between ϵ_t and asset class returns.

The objective function of this QP can be easily computed:

$$\begin{aligned} \mathrm{var}(R_t - w^T F_t) &= \frac{1}{T}\sum_{t=1}^{T}(R_t - w^T F_t)^2 - \left(\frac{\sum_{t=1}^{T}(R_t - w^T F_t)}{T}\right)^2 \\ &= \frac{1}{T}\|R - Fw\|^2 - \left(\frac{e^T(R - Fw)}{T}\right)^2 \\ &= \left(\frac{\|R\|^2}{T} - \frac{(e^T R)^2}{T^2}\right) - 2\left(\frac{R^T F}{T} - \frac{e^T R}{T^2}e^T F\right) w \\ &\quad + w^T\left(\frac{1}{T}F^T F - \frac{1}{T^2}F^T e e^T F\right) w. \end{aligned}$$

Above, we introduced and used the notation

$$R = \begin{bmatrix} R_1 \\ \vdots \\ R_T \end{bmatrix}, \text{ and } F = \begin{bmatrix} F_1 \\ \cdots \\ F_T \end{bmatrix} = \begin{bmatrix} F_{11} & \ldots & F_{n1} \\ \vdots & \ddots & \vdots \\ F_{1T} & \cdots & F_{nT} \end{bmatrix}$$

and e denotes a vector of ones of appropriate size. Convexity of this quadratic function of w can be easily verified. Indeed,

$$\frac{1}{T}F^T F - \frac{1}{T^2}F^T e e^T F = \frac{1}{T}F^T\left(I - \frac{ee^T}{T}\right)F, \tag{5.10}$$

and the symmetric matrix $M = I - \frac{ee^T}{T}$ in the middle of the right-hand-side expression above is a positive semidefinite matrix with only two eigenvalues: 0 (multiplicity 1) and 1 (multiplicity $T-1$). Since M is positive semidefinite, so is $F^T MF$ and therefore the variance of ϵ_t is a convex quadratic function of w. Therefore, the problem (5.9) is convex quadratic programming problem and is easily solvable using well-known optimization techniques such as interior-point methods.

5.4 Recovering Risk-Neural Probabilities from Options Prices

Recall our discussion on risk-neutral probability measures in Section 3.1.2. There, we considered a one-period economy with are n securities. Current prices of these securities are denoted by S_0^i for $i = 1, \ldots, n$. At the end of the current period, the economy will be in one of the states from the state space Ω. If the economy reaches state $\omega \in \Omega$ at the end of the current period, security i will have the payoff $S_1^i(\omega)$. We assume that we know all S_0^i's and $S_1^i(\omega)$'s but do not know the particular terminal state ω, which will be determined randomly.

Let r denote the one-period (riskless) interest rate and let $R = 1 + r$. A risk neutral probability measure (RNPM) is defined as the probability measure under which the present value of the expected value of future payoffs of a security equals its current price. More specifically,

- (**discrete case:**) on the state space $\Omega = \{\omega_1, \omega_2, \ldots, \omega_m\}$, an RNPM is a vector of positive numbers $p_1, p_2, \ldots, p_m$ such that
  1. $\sum_{j=1}^m p_j = 1$,
  2. $S_0^i = \frac{1}{R} \sum_{j=1}^m p_j S_1^i(\omega_j),\ \forall i$.
- (**continuous case:**) on the state space $\Omega = (a, b)$ an RNPM is a density function $p : \Omega \to \Re_+$ such that
  1. $\int_a^b p(\omega) d\omega = 1$,
  2. $S_0^i = \frac{1}{R} \int_a^b p(\omega) S_1^i(\omega) d\omega,\ \forall i$.

Also recall the following result from Section 3.1.2 that is often called the *First Fundamental Theorem of Asset Pricing*:

Theorem 5.1 *A risk-neutral probability measure exists if and only if there are no arbitrage opportunities.*

If we can identify a risk-neutral probability measure associated with a given state space and a set of observed prices we can price any security for which we can determine

the payoffs for each state in the state space. Therefore, a fundamental problem in asset pricing is the identification of a RNPM consistent with a given set of prices. Of course, if the number of states in the state space is much larger than the number of observed prices, this problem becomes under-determined and we can not obtain a unique or sensible solution without introducing some additional structure into the RNPM we seek. In this section, we outline a strategy that guarantees the smoothness of the RNPM by constructing it through cubic splines. We first describe spline functions briefly:

Consider a function $f : [a, b] \to \Re$ to be estimated using its values $f_i = f(x_i)$ given on a set of points $\{x_i\}$, $i = 0, \ldots, m$. It is assumed that $x_0 = a$ and $x_m = b$.

A **spline function**, or **spline**, is a *piecewise* polynomial approximation $S(x)$ to the function f such that the approximation agrees with f on each node x_i, i.e., $S(x_i) = f(x_i), \forall i$.

The graph of a spline function S contains the data points (x_i, f_i) (called *knots*) and is continuous on $[a, b]$.

A spline on $[a, b]$ is of order n if (i) its first $n - 1$ derivatives exist on each interior knot, (ii) the highest degree for the polynomials defining the spline function is n.

A cubic (third order) spline uses cubic polynomials of the form $f_i(x) = \alpha_i x^3 + \beta_i x^2 + \gamma_i x + \delta_i$ to estimate the function in each interval $[x_{i-1}, x_i]$ for $i = 1, \ldots, m$. A cubic spline can be constructed in such a way that it has second derivatives at each node. For $m+1$ knots $(x_0, \ldots x_m)$ there are m intervals and, therefore $4m$ unknown constants to evaluate. To determine these $4m$ constants we use the following $4m$ equations:

1. $f_i(x_i) = f(x_i), i = 1, \ldots (n-1)$. $(n-1$ eqns$)$
2. $f_1(a) = f(a)$ and $f_m(b) = f(b)$. (2 eqns)
3. $f_i(x_i) = f_{i+1}(x_i), i = 1, \ldots (n-1)$. $(n-1$ eqns$)$
4. $f_i'(x_i) = f_{i+1}'(x_i), i = 1, \ldots (n-1)$. $(n-1$ eqns$)$
5. $f_i''(x_i) = f_{i+1}''(x_i), i = 1, \ldots (n-1)$. $(n-1$ eqns$)$
6. $f_1''(a) = 0$ and $f_m''(b) = 0$. (2 eqns)

The last condition leads to a so-called *natural* spline that is linear at both ends.

We now formulate a quadratic programming problem with the objective of finding a risk-neutral probability density function (described by cubic splines) for future values of an underlying security that fits the observed option prices on this security.

We fix the security under consideration, say a stock or an index. We also a fix an exercise date–this is the date for which we will obtain a pdf of the price of our security. Finally, we fix a range $[a, b]$ for possible terminal values of the price of the underlying security at the exercise date of the options and an interest rate r for the period between now and the exercise date. The inputs to our optimization problem

are current market prices C_K of call options and P_K for put options on the chosen underlying security with strike price K and the chosen expiration date. This data is freely available from newspapers and the Internet. Let $\mathcal{C}$ and $\mathcal{P}$, respectively, denote the set of strike prices K for which reliable market prices C_K and P_K are available. For example, $\mathcal{C}$ may denote the strike prices of call options that were traded on the day the problem is formulated.

Next, we fix a super-structure for the spline approximation to the risk-neutral density, meaning that we choose how many knots to use, where to place the knots and what kind of polynomial (quadratic, cubic, etc.) functions to use. For example, we may decide to use cubic splines and $m+1$ equally spaced knots. The parameters of the polynomial functions that comprise the spline function will be the variables of the optimization problem we are formulating. For cubic splines with $m+1$ knots, we will have $4m$ variables $(\alpha_i, \beta_i, \gamma_i, \delta_i)$ for $i = 1, \ldots, m$. Collectively, we will represent these variables with y. For all y chosen so that the corresponding polynomial functions f_i satisfy the equations 3–6 above, we will have a particular choice of a natural spline function defined on the interval $[a, b]$[1]. Let $p_y(\cdot)$ denote this function. Imposing the following additional restrictions we make sure that p_y is a probability density function:

$$p_y(x) \geq 0, \forall x \in [a, b] \tag{5.11}$$

$$\int_a^b p_y(\omega) d\omega = 1. \tag{5.12}$$

Next, we define the discounted expected value of the terminal value of each option using p_y as the risk-neutral density function:

$$C_K(y) := \frac{1}{1+r} \int_a^b (\omega - K)^+ p_y(\omega) d\omega, \tag{5.13}$$

$$P_K(y) := \frac{1}{1+r} \int_a^b (K - \omega)^+ p_y(\omega) d\omega. \tag{5.14}$$

Then,

$$(C_K - C_K(y))^2$$

measures the difference between the actual and theoretical values of the option if S_y was the actual RNPM. Now consider the aggregated error function for a given y:

$$E(y) := \sum_{K \in \mathcal{C}} (C_K - C_K(y))^2 + \sum_{K \in \mathcal{P}} (P_K - P_K(y))^2$$

The objective now is to choose y such that conditions 3–6 of spline function description as well as (5.11)–(5.12) are satisfied and $E(y)$ is minimized. This is essentially a constrained least squares problem and we can ensure that $E(y)$ is a convex quadratic function of y using the following strategy.

[1]Note that we do not impose the conditions 1 and 2, because the values of the pdf we are approximating are unknown and will be determined as a solution of an optimization problem.

We choose the number of knots and their locations so that the knots form a superset of $\mathcal{C} \cup \mathcal{P}$. Let $x_0 = a, x_1, \ldots, x_m = b$ denote the locations of the knots. Now, consider a call option with strike K and assume that K coincides with the location of the jth knot, i.e., $x_j = K$. Recall that y denotes collection of variables $(\alpha_i, \beta_i, \gamma_i, \delta_i)$ for $i = 1, \ldots, m$. Now, we can derive a formula for $C_K(y)$:

$$\begin{aligned}
(1+r)C_K(y) &= \int_a^b S_y(\omega)(\omega - K)^+ d\omega \\
&= \sum_{i=1}^{m} \int_{x_{i-1}}^{x_i} S_y(\omega)(\omega - K)^+ d\omega \\
&= \sum_{i=j+1}^{m} \int_{x_{i-1}}^{x_i} S_y(\omega)(\omega - K) d\omega \\
&= \sum_{i=j+1}^{m} \int_{x_{i-1}}^{x_i} \left(\alpha_i \omega^3 + \beta_i \omega^2 + \gamma_i \omega + \delta_i\right)(\omega - K) d\omega.
\end{aligned}$$

It is easily seen that this expression for $C_K(y)$ is a linear function of the components $(\alpha_i, \beta_i, \gamma_i, \delta_i)$ of the y variable. A similar formula can be derived for $P_K(y)$. The reason for choosing the knots at the strike prices is the third equation in the sequence above—we can immediately ignore some of the terms in the summation and the $(\cdot)^+$ function is linear (and not piecewise linear) in each integral.

Now, it is clear that the problem of minimizing $E(y)$ subject to spline function conditions and (5.11)–(5.12) is a quadratic optimization problem[2].

5.5 Exercises

1. Recall the mean-variance optimization problem we considered in Section 5.1:

$$\begin{array}{lrcl}
\min_x & x^T Q x & & \\
& \mu^T x & \geq & R \\
& Ax & = & b \\
& Cx & \geq & d.
\end{array} \tag{5.15}$$

   Now, consider the problem of finding the feasible portfolio with smallest overall variance, without imposing any expected return constraint:

$$\begin{array}{lrcl}
\min_x & x^T Q x & & \\
& Ax & = & b \\
& Cx & \geq & d.
\end{array} \tag{5.16}$$

[2]The formulation we outlined above does not guarantee that the spline approximation willbe non-negative in its domain and therefore can result in an inproper solution since we are estimating a probability distribution function. If this occurs, one could subdivide the intervals where the spline functions become negative by introducing additional knots and continue in this manner until a non-negative approximation is obtained .

(i) Does the optimal solution to (5.16) give an efficient portfolio? Why?

(ii) Let x_R, $\lambda_R \in \Re$, $\gamma_E \in \Re^m$, and $\gamma_I \in \Re^p$ satisfy the optimality conditions of (5.15) (see system (5.2)). If $\lambda_R = 0$, show that x_R is an optimal solution to (5.16). (Hint: What are the optimality conditions for (5.16)? How are they related to (5.2)?)

2. Implement the returns-based style analysis approach to determine the effective asset mix of your favorite mutual fund. Use the following asset classes as your "factors": Large growth stocks, large value stocks, small growth stocks, small value stocks, international stocks, and fixed income investments. You should obtain time series of returns representing these asset classes from on-line resources. You should also obtain a corresponding time series of returns for the mutual fund you picked for this exercise. Solve the problem using 30 periods of data (i.e., $T = 30$).

3. Classification problems are among the important classes of problems in financial mathematics that can be solved using optimization models and techniques. In a classification problem we have a vector of "feature"s describing an entity and the objective is to analyze the features to determine which one of the two (or more) "classes" each entity belongs to. For example, the classes might be "growth stocks" and "value stocks", and the entities (stocks) may be described by a feature vector that may contain elements such as stock price, price-earnings ratio, growth rate for the previous periods, growth estimates, etc.

   Mathematical approaches to classification often start with a "training" exercise. One is supplied with a list of entities, their feature vectors and the classes they belong to. From this information, one tries to extract a mathematical structure for the entity classes so that additional entities can be classified using this mathematical structure and their feature vectors. For two-class classification, a hyperplane is probably the simplest mathematical structure that can be used to "separate" the feature vectors of these two different classes. Of course, a hyperplane is often not sufficient to separate two sets of vectors, but there are certain situations it may be sufficient.

   Consider feature vectors $a_i \in \Re^n$ for $i = 1, \ldots, k_1$ corresponding to class 1, and vectors $b_i \in \Re^n$ for $i = 1, \ldots, k_2$ corresponding to class 2. If these two vector sets can be linearly separated, there exists a hyperplane $w^T x = \gamma$ with $w \in \Re^n, \gamma \in \Re$ such that

$$\begin{aligned} w^T a_i &\geq \gamma, \text{ for } i = 1, \ldots, k_1 \\ w^T b_i &\leq \gamma, \text{ for } i = 1, \ldots, k_2. \end{aligned}$$

   To have a "strict" separation, we often prefer to obtain w and γ such that

$$\begin{aligned} w^T a_i &\geq \gamma + 1, \text{ for } i = 1, \ldots, k_1 \\ w^T b_i &\leq \gamma - 1, \text{ for } i = 1, \ldots, k_2. \end{aligned}$$

In this manner, we find two parallel lines ($w^T x = \gamma + 1$ line and $w^T x = \gamma - 1$) that form the boundary of the class 1 and class 2 portion of the vector space. There may be several such parallel lines that separate the two classes. Which one should one choose? A good criterion is to choose the lines that have the largest margin (distance between the lines).

a) Consider the following quadratic problem:

$$\begin{array}{lrcl} \min_{w,\gamma} & \|w\|_2^2 & & \\ & a_i^T w & \geq & \gamma + 1, \text{ for } i = 1, \ldots, k_1 \\ & b_i^T w & \leq & \gamma - 1, \text{ for } i = 1, \ldots, k_2. \end{array} \tag{5.17}$$

Show that the objective function of this problem is equivalent to maximizing the margin between the lines $w^T x = \gamma + 1$ and $w^T x = \gamma - 1$.

b) The linear separation idea we presented above can be used even when the two vector sets $\{a_i\}$ and $\{b_i\}$ are not linearly separable. (Note that linearly inseparable sets will result in an infeasible problem in formulation (5.17).) This is achieved by introducing a nonnegative "violation" variable for each constraint of (5.17). Then, one has two objectives: to minimize the total of the violations of the constraints of (1) and to maximize the margin. Develop a quadratic programming model that combines these two objectives using an adjustable parameter that can be chosen in a way to put more weight on violations or margin, depending on one's preference.

4. The classification problems we discussed in the previous exercise can also be formulated as linear programming problems, if one agrees to use 1-norm rather than 2-norm of w in the objective function. Recall that $\|w\|_1 = \sum_i |w_i|$. Show that if we replace $\|w\|_2^2$ with $\|w\|_1$ in the objective function of (1), we can write the resulting problem as an LP. Show also that, this new objective function is equivalent to maximizing the distance between $w^T x = \gamma + 1$ and $w^T x = \gamma - 1$ if one measures the distance using ∞-norm ($\|g\|_\infty = \max_i |g_i|$).

Chapter 6

Stochastic Programming Models

6.1 Introduction to Stochastic Programming

In the Introduction and elsewhere, we argued that many optimization problems are described by uncertain parameters. When these uncertain parameters can be considered as random variables and have known probability distributions, new optimization problems can be formulated that involve expected values of these random variables. In this manner, one obtains a new problem (called the *deterministic equivalent*) that removes the uncertainty and results in a deterministic optimization problem. This is the approach of *stochastic programming*.

While stochastic programming models have existed for several decades, computational technology has only recently allowed the solution of realistic size problems. The field continues to develop with the advancement of available algorithms and computational power. It is a popular modeling tool for problems in a variety of disciplines including financial engineering.

In analogy to the generic optimization problem ($\mathcal{OP}$) we considered in the Introduction, a generic stochastic programming problem can be formulated as follows:

$$(\mathcal{SP}) \qquad \begin{array}{rrcl} \min_x & E[f(x,p)] & & \\ & E[g_i(x,p)] & = & 0, i \in \mathcal{E} \\ & E[g_i(x,p)] & \geq & 0, i \in \mathcal{I}, \\ & x \in S. & & \end{array} \tag{6.1}$$

In this formulation, x represents our n-dimensional decision variable vector and p represents the uncertain parameters of the optimization problem. As we mentioned, in the stochastic programming approach we assume that uncertain parameters are random, therefore, p is a random vector. Let k denote the dimension of p and let P be its probability distribution function on $\Re^k$.

S represents (implicitly) the constraints that do not depend on the random parameters p. g_i's are the equality and inequality constraints that depend on p. In contrast

to $(\mathcal{OP})$, the functions f and g_i in (6.1) are from $\Re^{n+k}$ to $\Re$ (instead of being from $\Re^n$ to $\Re$). However, the expectation functionals are from $\Re^n$ to $\Re$:

$$\begin{aligned}(Ef)(x) &:= E[f(x,p)] = \int_{\Re^k} f(x,p)dP(p),\\ (Eg_i)(x) &:= E[g_i(x,p)] = \int_{\Re^k} g_i(x,p)dP(p).\end{aligned}$$

Stochastic programming models can include both *anticipative* and *adaptive* decision variables. Anticipative variables correspond to those decisions that must be made *here-and-now* and can not depend on the future observations/partial realizations of the random parameters. Adaptive variables correspond to *wait-and-see* decisions that can be made after some (or, sometimes all) of the random parameters are observed.

When a stochastic programming model involves only anticipative variables and the constraints depend on random parameters, feasibility of the variables is sometimes described using *chance constraints* that impose lower limits on the satisfaction probability of the constraint. For example, a chance constraint may take the following form:

$$P\{g_i(x,p) \geq 0\} \geq \theta_i.$$

Of course, many realistic models include both anticipative and adaptive variables. *Recourse* models bring these two types of variables together. Using a multi-stage stochastic programming formulation, with recourse variables at each stage, one can model a decision environment where information is revealed progressively and the decisions are adapted to each new piece of information.

In investment planning, each new trading opportunity represents a new decision to be made. Therefore, trading dates where investment portfolios can be rebalanced become natural choices for decision stages, and these problems can be formulated conveniently as multi-stage stochastic programming problems with recourse.

6.2 Two Stage Problems with Recourse

The next three sections are based on the Stochastic Programming chapter in the course notes by Gerard Cornuejols for course "Optimization Methods in Finance". In the Introduction, we have already seen a generic form of a *two-stage stochastic linear program with recourse*. We follow the standard notation for stochastic LPs here which is slightly different from that of $(\mathcal{SP})$:

$$\begin{array}{lrcrcl} \max & (c^1)^T x^1 & + & E[\max c^2(\omega)^T x^2(\omega)] & & \\ & A^1 x^1 & & & = & b^1 \\ & B^2(\omega)x^1 & + & A^2(\omega)x^2(\omega) & = & b^2(\omega) \\ & x^1 \geq 0, & & x^2(\omega) \geq 0. & & \end{array} \tag{6.2}$$

Above, the first-stage decisions are represented by vector x^1 and the second-stage decisions by vector $x^2(\omega)$, which depend on the realization ω of a random event. A^1

and b^1 define deterministic constraints on the first-stage decisions x^1, whereas $A^2(\omega)$, $B^2(\omega)$, and $b^2(\omega)$ define stochastic linear constraints linking the recourse decisions $x^2(\omega)$ to the first-stage decisions. The objective function contains a deterministic term $(c^1)^T x^1$ and the expectation of the second-stage objective $c^2(\omega)^T x^2(\omega)$ taken over all realizations of the random event ω.

Notice that the first-stage decisions will not necessarily satisfy the linking constraints $B^2(\omega)x^1 + A^2(\omega)x^2(\omega) = b^2(\omega)$, if no recourse action is taken. Therefore, recourse allows one to make sure that the initial decisions can be "corrected" with respect to this second set of feasibility equations.

In Section 1.2.1, we also argued that problem (6.2) can be represented in an alternative manner by considering the *second-stage* or *recourse* problem that is defined as follows, given x^1, the first-stage decisions:

$$\begin{array}{rrcl} f(x^1,\omega) = \max & c^2(\omega)^T x^2(\omega) & & \\ & A^2(\omega)x^2(\omega) & = & b^2(\omega) - B^2(\omega)x^1 \\ & x^2(\omega) \geq 0, & & \end{array} \tag{6.3}$$

Now, the alternative formulation of the two-stage stochastic linear program is given as

$$\begin{array}{ll} \max & (c^1)^T x^1 + E[f(x^1,\omega)] \\ & A^1 x^1 = b^1 \\ & x^1 \geq 0, \\ & f(x^1,\omega) = \max_{x^2(\omega)}\{c^2(\omega)^T x^2(\omega) | A^2(\omega)x^2(\omega) = b^2(\omega) - B^2(\omega)x^1, x^2(\omega) \geq 0\}. \end{array} \tag{6.4}$$

Next, we consider the case where the sample space Ω for the random event vector ω is a finite set. Assume that $\Omega = \{\omega_1, \ldots, \omega_{K_2}\}$ and let $p = (p_1, \ldots, p_{K_2})$ denote the probability distribution on this sample space. For brevity, we write c^2_k instead of $c^2(\omega_k)$, etc. Under this *scenario approach* the two-stage stochastic linear programming problem takes the following form:

$$\begin{array}{ll} \max & (c^1)^T x^1 + \sum_{k_2=1}^{K_2} p_{k_2} f(x^1,\omega_{k_2})] \\ & A^1 x^1 = b^1 \\ & x^1 \geq 0, \\ & f(x^1,\omega_{k_2}) = \max_{x^2_{k_2}}\{(c^2_{k_2})^T x^2_{k_2} | A^2_j x^2_{k_2} = b^2_{k_2} - B^2_{k_2} x^1, x^2_{k_2} \geq 0\}, k_2 = 1, \ldots, K_2. \end{array} \tag{6.5}$$

The maximum in the objective will be achieved if and only if all second-stage problems achieve their maximum objective values. Therefore, we can eliminate the $f(x_1,\omega_{k_2})$ expressions and get the following problem:

$$\begin{array}{rrcrcccrcl} \max & (c^1)^T x^1 & + & p_1(c^2_1)^T x^2_1 & + & \ldots & + & p_k(c^2_{K_2})^T x^2_{K_2} & & \\ & A^1 x^1 & & & & & & & = & b^1 \\ & B^2_j x^1 & + & A^2_1 x^2_1 & & & & & = & b^2_1 \\ & \vdots & & & \ddots & & & & & \vdots \\ & B^2_{K_2} x^1 & & & & & + & A^2_{K_2} x^2_{K_2} & = & b^2_{K_2} \\ & x^1, & & x^2_1, & & \ldots & & x^2_{K_2} & \geq & 0. \end{array} \tag{6.6}$$

This is a deterministic linear programming problem called the *deterministic equivalent* of the original uncertain problem. This problem has k copies of the second-stage decision variables and therefore, can be significantly larger than the original problem before we considered the uncertainty of the parameters. Fortunately, however, the constraint matrix has a very special sparsity structure that can be exploited by modern decomposition based solution methods.

6.3 Multi Stage Problems

A *multi-stage stochastic program with recourse* is formulated as follows:

$$\begin{array}{rllllll}
\max & c^1x^1 + E_{\omega^2}[\max\,(c^2(\omega^2)x^2(\omega^2) + \ldots + E_{\omega^T|\omega^{T-1}}[c^T(\omega^T)x^T(\omega^T)]\ldots)] & & & & & \\
\text{subject to} & & & & & & \\
& A^1x^1 & & & & = & b^1 \\
& B^2(\omega^2)x^1 & +A^2(\omega^2)x^2(\omega^2) & & & = & b^2(\omega^2) \\
& & B^3(\omega^3)x^2(\omega^2) & +A^3(\omega^3)x^3(\omega^3) & & = & b^3(\omega^3) \\
& & & \cdots & & & \\
& & & B^T(\omega^T)x^{T-1}(\omega^{T-1}) & +A^T(\omega^T)x^T(\omega^T) & = & b^T(\omega^T) \\
& x^1 \geq 0, & x^2(\omega^2) \geq 0, & \cdots & x^T(\omega^T) \geq 0 & &
\end{array} \tag{6.7}$$

where the first-stage decisions are represented by vector x^1 and the stage t decisions by vector $x^t(\omega^t)$, for $t = 2, \ldots, T$. In the objective, $E_{\omega^t|\omega^{t-1}}$ denotes the conditional expectation of the state ω^t of the data process $\omega = (\omega^2, \ldots, \omega^T)$ at time t given the history of the process up to time $t-1$. The sequence of events is the following:

$$\text{decide } x^1 \quad \text{observe } \omega^2 \quad \text{decide } x^2(\omega^2) \quad \ldots \quad \text{observe } \omega^T \quad \text{decide } x^T(\omega^T).$$

Using the approach presented in Section 1.2.1 and above for the two-stage case, one can reduce multistage stochastic programs to a nested sequence of nonlinear programs. Alternatively, we can again consider the *scenario approach.* In the multi-stage case, one can think of the scenarios as a tree with T levels where, at each stage $t = 1, \ldots, T-1$, the branches correspond to refinements of ω^t to ω^{t+1}. The stochastic program can then be written as a very large linear program as follows. For $t = 2, \ldots, T$, let $p^{k_2,k_3,\ldots,k_t}$ denote the conditional probability of outcome k_t at stage t given outcomes $k_2, \ldots, k_{t-1}$ at the previous stages. Then (6.7) can be written as

$$
\begin{array}{rl}
\max & c^1x^1 + \sum_{k_2=1}^{K_2}[p^{k_2}(c^{k_2}x^{k_2} + \ldots + \sum_{k_T=1}^{K_T}[p^{k_2,\ldots,k_T}c^{k_2,\ldots,k_T}x^{k_2,\ldots,k_T}]\ldots)] \\
\text{subject to} & \\
& \begin{array}{rrrrrcl}
A^1x^1 & & & & & = & b^1 \\
B^{k_2}x^1 & +A^{k_2}x^{k_2} & & & & = & b^{k_2} \\
 & B^{k_2k_3}x^{k_2} & +A^{k_2k_3}x^{k_2k_3} & & & = & b^{k_2k_3} \\
 & & & \ldots & & & \\
 & & B^{k_2,\ldots,k_T}x^{k_2,\ldots,k_{T-1}} & +A^{k_2,\ldots,k_T}x^{k_2,\ldots,k_T} & = & b^{k_2,\ldots,k_T} \\
x^1 \geq 0, & x^{k_2} \geq 0, & \ldots & x^{k_2,\ldots,k_T} \geq 0 & &
\end{array}
\end{array}
\tag{6.8}
$$

where

$$
B^{k_2} := \begin{pmatrix} B^2(\omega_1^2) \\ B^2(\omega_2^2) \\ \ldots \\ B^2(\omega_{K_2}^2) \end{pmatrix},
$$

the matrix obtained by stacking all the possible realizations of $B^2(\omega^2)$. The matrices A^{k_2} and b^{k_2} are defined similarly. The matrix $B^{k_2k_3}$ is defined as

$$
\begin{pmatrix}
B^3(\omega_{1,1}^3) & 0 & \ldots & 0 \\
\ldots & \ldots & \ldots & \ldots \\
B^3(\omega_{1,K_3}^3) & 0 & \ldots & 0 \\
0 & B^3(\omega_{2,1}^3) & \ldots & 0 \\
\ldots & \ldots & \ldots & \ldots \\
0 & B^3(\omega_{2,K_3}^3) & \ldots & 0 \\
\ldots & \ldots & \ldots & \ldots \\
0 & 0 & \ldots & B^3(\omega_{K_2,1}^3) \\
\ldots & \ldots & \ldots & \ldots \\
0 & 0 & \ldots & B^2(\omega_{K_2,K_3}^3)
\end{pmatrix},
$$

where ω_{ij}^3 for $i = 1, \ldots, K_2$ and $j = 1, \ldots, K_3$ are all the possible realizations of ω^3. The remaining matrices are defined similarly. It should be clear from this construction that (6.8) is a very large scale linear program. Indeed, if $B^i(\omega^i)$ has m_i rows and n_i columns, then $B^{k_2,\ldots,k_i}$ has $m_iK_2K_3\ldots K_i$ rows and $n_iK_2K_3\ldots K_{i-1}$ columns. For example, for a problem with 10 stages and a binary tree, there are 1024 scenarios and therefore (6.8) has several thousand constraints and variables. Modern commercial codes can handle such large linear programs, but a moderate increase in the number of stages or in the number of branches at each stage could make (6.8) too large to solve.

6.4 Stochastic Programming Models and Tools in Finance

6.4.1 Asset/Liability Management

Financial health of any company, and in particular those of financial institutions, is reflected in the balance sheets of the company. Proper management of the company requires attention to both sides of the balance sheet–assets and liabilities. Asset-liability management (ALM) offers sophisticated mathematical tools for an integrated management of assets and liabilities and is the focus of many studies in financial mathematics.

ALM recognizes that static, one period investment planning models (such as mean-variance optimization) fail to incorporate the multi-period nature of the liabilities faced by the company. A multi-period model that emphasizes the need to meet liabilities in each period for a finite (or possibly infinite) horizon is often required. Since liabilities and asset returns usually have random components, their optimal management requires tools of "Optimization under Uncertainty" and most notably, stochastic programming approaches.

We recall the ALM setting we introduced in Section 1.3.4: Let L_t be the liability of the company in year t for $t = 1, \ldots, T$. The L_t's are random variables. Given these liabilities, which assets (and in which quantities) should the company hold each year to maximize its expected wealth in year T? The assets may be domestic stocks, foreign stocks, real estate, bonds, etc. Let R_{it} denote the return on asset i in year t. The R_{it}'s are random variables. The decision variables are:

$$x_{it} = \text{ market value invested in asset } i \text{ in year } t.$$

The decisions x_{it} in year t are made after the random variables L_t and R_{it} are realized. That is, the decision problem is multistage, stochastic, with recourse. The stochastic program can be written as follows.

$$\begin{array}{rl} \max & E[\sum_i x_{iT}] \\ \text{subject to} & \\ \text{asset accumulation:} & \sum_i (1+R_{it}) x_{i,t-1} - \sum_i x_{it} = L_t \quad \text{for } t = 1, \ldots, T \\ & x_{it} \geq 0. \end{array}$$

The constraint says that the surplus left after liability L_t is covered will be invested as follows: x_{it} invested in asset i. In this formulation, $x_{0,t}$ are the fixed, and possibly nonzero initial positions in different asset classes. The objective selected in the model above is to maximize the expected wealth at the end of the planning horizon. In practice, one might have a different objective. For example, in some cases, minimizing Value at Risk (VaR) might be more appropriate. Other priorities may dictate other objective functions.

To address the issue of the most appropriate objective function, one must understand the role of liabilities. Pension funds and insurance companies are among the most

typical arenas for the integrated management of assets and liabilities through ALM. We consider the case of a Japanese insurance company, the Yasuda Fire and Marine Insurance Co, Ltd. In this case, the liabilities are mainly savings-oriented policies issued by the company. Each new policy sold represents a deposit, or inflow of funds. Interest is periodically credited to the policy until maturity, typically three to five years, at which time the principal amount plus credited interest is refunded to the policyholder. The crediting rate is typically adjusted each year in relation to a market index like the prime rate. Therefore, we cannot say with certainty what future liabilities will be. Insurance business regulations stipulate that interest credited to some policies be earned from investment income, not capital gains. So, in addition to ensuring that the maturity cash flows are met, the firm must seek to avoid interim shortfalls in income earned versus interest credited. In fact, it is the risk of not earning adequate income quarter by quarter that the decision makers view as the primary component of risk at Yasuda.

The problem is to determine the optimal allocation of the deposited funds into several asset categories: cash, fixed rate and floating rate loans, bonds, equities, real estate and other assets. Since we can revise the portfolio allocations over time, the decision we make is not just among allocations today but among allocation strategies over time. A realistic dynamic asset/liability model must also account for the payment of taxes. This is made possible by distinguishing between income return and price return.

A stochastic linear program as in (6.7) is used to model the problem. The linear program has uncertainty in many coefficients. This uncertainty is modeled through a finite number of scenarios. In this fashion, the problem is transformed into a very large scale linear program of the form (6.8). The random elements include price and income returns for each asset class, as well as policy crediting rates.

We now present a multistage stochastic program that was developed for The Yasuda Fire and Marine Insurance Co., Ltd. Our presentation follows the description of the model as stated in [3].

Stages are indexed by $t = 0, 1, \ldots, T$.

Decision variables of the stochastic program:

$$\begin{aligned} x_{it} &= \text{market value in asset } i \text{ at } t \\ w_t &= \text{income shortfall at } t \geq 1 \\ v_t &= \text{income surplus at } t \geq 1 \end{aligned}$$

Random variables appearing in the stochastic linear program: For $t \geq 1$,

$$\begin{aligned} RP_{it} &= \text{price return of asset } i \text{ from } t-1 \text{ to } t \\ RI_{it} &= \text{income return of asset } i \text{ from } t-1 \text{ to } t \\ F_t &= \text{deposit inflow from } t-1 \text{ to } t \\ P_t &= \text{principal payout from } t-1 \text{ to } t \end{aligned}$$

$$\begin{aligned} I_t &= \text{interest payout from } t-1 \text{ to } t \\ g_t &= \text{rate at which interest is credited to policies from } t-1 \text{ to } t \\ L_t &= \text{liability valuation at } t \end{aligned}$$

Parametrized function appearing in the objective:

$$c_t = \text{piecewise linear convex cost function}$$

The objective of the model is to allocate funds among available assets to maximize expected wealth at the end of the planning horizon T less expected penalized shortfalls accumulated through the planning horizon.

$$\begin{array}{rl} \max & E[\sum_i x_{iT} - \sum_{t=1}^{T} c_t(w_t)] \\ \text{subject to} & \\ \text{asset accumulation:} & \sum_i x_{it} - \sum_i (1 + RP_{it} + RI_{it}) x_{i,t-1} = F_t - P_t - I_t \quad \text{for } t = 1, \ldots, T \\ \text{income shortfall:} & \sum_i RI_{it} x_{i,t-1} + w_t - v_t = g_t L_{t-1} \quad \text{for } t = 1, \ldots, T \\ & x_{it} \geq 0, \quad w_t \geq 0, \quad v_t \geq 0. \end{array} \tag{6.9}$$

Liability balances and cash flows are computed so as to satisfy the liability accumulation relations.

$$L_t = (1 + g_t) L_{t-1} + F_t - P_t - I_t \quad \text{for } t \geq 1.$$

The stochastic linear program (6.9) is converted into a large linear program using a finite number of scenarios to deal with the random elements in the data. Creation of scenario inputs is made in stages using a tree. The tree structure can be described by the number of branches at each stage. For example, a 1-8-4-4-2-1 tree has 256 scenarios. Stage $t = 0$ is the initial stage. Stage $t = 1$ may be chosen to be the end of Quarter 1 and has 8 different scenarios in this example. Stage $t = 2$ may be chosen to be the end of Year 1, with each of the previous scenarios giving rise to 4 new scenarios, and so on. For the Yasuda Fire and Marine Insurance Co., Ltd., a problem with 7 asset classes and 6 stages gives rise to a stochastic linear program (6.9) with 12 constraints (other than nonnegativity) and 54 variables. Using 256 scenarios, this stochastic program is converted into a linear program with several thousand constraints and over 10,000 variables. Solving this model yielded extra income estimated to about US$ 80 million per year for the company.

6.4.2 Corporate Debt Management

A closely related problem to the asset-liability management (ALM) problem in corporate financial planning is the problem of debt management. Here the focus is on retiring (paying back) outstanding debt at minimum cost. More specifically, corporate

debt managers must make financial decisions to minimize the costs and risks of borrowing to meet debt financing requirements. These requirements are often determined by the firm's investment decisions. Our discussion in this subsection is based on the article [4].

Debt managers need to choose the sources of borrowing, types of debts to be used, timing and terms of debts, whether the debts will be callable[1], etc., in a multi-period framework where the difficulty of the problem is compounded by the fact that the interest rates that determine the cost of debt are uncertain. Since interest rate movements can be modeled by random variables this problem presents an attractive setting for the use of stochastic programming techniques. Below, we discuss a deterministic linear programming equivalent of stochastic LP model for the debt management problem.

We consider a multi-period framework with T time periods. We will use the indices s and t ranging between 0 (now) and T (termination date, or horizon) to denote different time periods in the model. We consider K types of debt that are distinguished by market of issue, term and the presence (or absence) of call option available to the borrower. In our notation, the superscript k ranging between 1 and K will denote the different types of debt being considered.

The evolution of the interest rates are described using a scenario tree. We denote by $e_j = e_{j1}, e_{j2}, \ldots, e_{jT}, j = 1, \ldots, J$ a sample path of this scenario tree which corresponds to a sequence of interest rate events. When a parameter or variable is contingent on the event sequence e_j we use the notation (e_j) (see below).

The decision variables in this model are the following:

- $B_t^k(e_j)$: dollar amount at par[2] of debt type k **B**orrowed at the beginning of period t.
- $O_{s,t}^k(e_j)$: dollar amount at par of debt type k borrowed in period s and **O**utstanding at the beginning of period t.
- $R_{s,t}^k(e_j)$: dollar amount at par of debt type k borrowed in period s and **R**etired (paid back) at the beginning of period t.
- $S_t(e_j)$: dollar value of **S**urplus cash held at the beginning of period t.

Next, we list the input parameters to the problem:

- $r_{s,t}^k(e_j)$: interest payment in period t per dollar outstanding of debt type k issued in period s.
- f_t^k: issue costs (excluding premium or discount) per dollar borrowed of debt type k issued in period t.

[1] A *callable debt* is a debt security whose issuer has the right to redeem the security prior to its stated maturity date at a price established at the time of issuance, on or after a specified date.

[2] At a price equal to the par (face) value of the security; the original issue price of a security.

- $g^k_{s,t}(e_j)$: retirement premium or discount per dollar for debt type k issued in period s, if retired in period t[3].

- $i_t(e_j)$: interest earned per dollar on surplus cash in period t.

- $p(e_j)$: probability of the event sequence e_j. Note that $p(e_j) \geq 0$, $\forall j$ and $\sum_{j=1}^{J} p(e_j) = 1$.

- C_t: cash requirements for period t, which can be negative to indicate an operating surplus.

- M_t: maximum allowable cost of debt service in period t.

- $q^k_t(Q^k_t)$: minimum (maximum) borrowing of debt type k in period t.

- $L_t(e_j)(U_t(e_j))$: minimum (maximum) dollar amount of debt (at par) retired in period t.

The objective function of this problem is expressed as follows:

$$\min \sum_{j=1}^{J} p(e_j) \left(\sum_{k=1}^{K} \sum_{t=1}^{T} \left(1 + g^k_{t,T}(e_j)\right) \left[O^k_{t,T}(e_j) - R^k_{t,T}(e_j)\right] + (1 - f^k_T) B^k_T(e_j) \right). \quad (6.10)$$

This function expresses the expected retirement cost of the total debt outstanding at the end of period T.

We complete the description of the deterministic equivalent of the stochastic LP by listing the constraints of the problem:

- **Cash Requirements:** For each time period $t = 1, \ldots, T$ and scenario path $j = 1, \ldots, J$:

$$C_t + S_t(e_j) = \sum_{k=1}^{K} \left\{ \left(1 - f^k_t\right) B^k_t(e_j) + \left(1 + i_{t-1}(e_j)\right) S_{t-1}(e_j) - \sum_{s=0}^{t-1} \left[r^k_{s,t}(e_j) O^k_{s,t}(e_j) - \left(1 + g^k_{s,t}(e_j)\right) R^k_{s,t}(e_j) \right] \right\}.$$

  This balance equation indicates that the difference between cash available (new net borrowing, surplus cash from previous period and the interest earned on this cash) and the debt payments (interest on outstanding debt and cash outflows on repayment) should equal the cash requirements plus the surplus cash left for this period.

[3]These parameters are used to define call options and to value the debt portfolio at the end of the planning period.

- **Debt Balance Constraints:** For $j = 1, \ldots, J$, $t = 1, \ldots, T$, $s = 0, \ldots, t-2$, and $k = 1, \ldots K$:

$$\begin{aligned} O^k_{s,t}(e_j) - O^k_{s,t-1}(e_j) + R^k_{s,t-1}(e_j) &= 0 \\ O^k_{t-1,t}(e_j) - B^k_{t-1}(e_j) - R^k_{t-1,t}(e_j) &= 0 \end{aligned}$$

- **Maximum cost of debt:** For $j = 1, \ldots, J$, $t = 1, \ldots, T$, and $k = 1, \ldots K$:

$$\sum_{s=1}^{t-1} \left(r^k_{s,t}(e_j) O^k_{s,t}(e_j) - i_{t-1}(e_j) S_{t-1}(e_j) \right) \leq M_t.$$

- **Borrowing limits:** For $j = 1, \ldots, J$, $t = 1, \ldots, T$, and $k = 1, \ldots K$:

$$q^k_t \leq B^k_t(e_j) \leq Q^k_t.$$

- **Payoff limits:** For $j = 1, \ldots, J$ and $t = 1, \ldots, T$:

$$L_t(e_j) \leq \sum_{k=1}^{K} \sum_{s=0}^{t-1} R^k_{s,t}(e_j) \leq U_t(e_j).$$

- **Nonnegativity:** For $j = 1, \ldots, J$, $t = 1, \ldots, T$, $s = 0, \ldots, t-2$, and $k = 1, \ldots K$:

$$B^k_t(e_j) \geq 0, \quad O^k_{s,t}(e_j) \geq 0, \quad R^k_{s,t}(e_j) \geq 0, \quad S_t(e_j) \geq 0.$$

In the formulation above, we used the notation of the article [4]. However, since the parameters and variables dependent on e_j can only depend on the portion of the sequence that is revealed by a certain time, a more precise notation can be obtained using the following ideas. First, let $e^t_j = e_{j1}, e_{j2}, \ldots, e_{jt}, j = 1, \ldots, J, t = 1, \ldots, T$, i.e., e^t_j represents the portion of e_j observed by time period t. Then, one replaces the expressions such as $S_t(e_j)$ with $S_t(e^t_j)$, etc.

Chapter 7

Robust Optimization Models and Tools in Finance

7.1 Introduction to Robust Optimization

Robust optimization refers to the modeling of optimization problems with data uncertainty to obtain a solution that is guaranteed to be "good" for all possible realizations of the uncertain parameters. As we argued in the introductory chapter, unlike the probabilistic descriptions of uncertainty, this approach gives the same importance to all possible realizations. Uncertainty in the parameters is described through *uncertainty sets* that contain all (or most) possible values that may be realized for the uncertain parameters.

Recall from our introduction that there are different definitions and interpretations of robustness and the resulting models differ accordingly. In particular, we distinguish between *model robustness* and *solution robustness*. In the first case, data uncertainty puts the feasibility of potential solutions at risk. In the second, feasibility constraints are fixed and the uncertainty of the objective function affects the proximity of the generated solutions to optimality. Next, we discuss each topic in detail.

7.2 Model Robustness

One of the most important concepts in robust optimization is *model robustness*. This refers to solutions that remain *feasible* for all possible values of the uncertain inputs–we will call such solutions *constraint-robust solutions*. This type of solutions are required in many engineering applications. Typical instances include multi-stage problems where the uncertain outcomes of earlier stages have an effect on the decisions of the later stages and the decision variables must be chosen to satisfy certain balance constraints (e.g., inputs to a particular stage can not exceed the outputs of the previous stage) no matter what happens with the uncertain parameters of the problem. Therefore, our

solution must be constraint-robust with respect to the uncertainties of the problem. Here is a mathematical model for finding constraint-robust solutions: Consider an optimization problem of the form:

$$(\mathcal{OP}_{uc}) \qquad \begin{array}{lll} \min_x & f(x) & \\ & G(x,p) & \in \quad K. \end{array} \tag{7.1}$$

Here, x are the decision variables, f is the (certain) objective function, G and K are the structural elements of the constraints that are assumed to be certain and p are the possibly uncertain parameters of the problem. Consider an uncertainty set $\mathcal{U}$ that contains all possible values of the uncertain parameters p. Then, a constraint-robust optimal solution can be found by solving the following problem:

$$(\mathcal{CROP}) \qquad \begin{array}{lll} \min_x & f(x) & \\ & G(x,p) & \in \quad K, \forall p \in \mathcal{U}. \end{array} \tag{7.2}$$

Above, $\mathcal{CROP}$ stands for *constraint-robust optimization problem.* Note that there are no uncertain parameters in the objective function of the problem $\mathcal{OP}_{uc}$. This, however, is not a restrictive assumption. An optimization problem with uncertain parameters in both the objective function and constraints can be easily reformulated to fit the form in $\mathcal{OP}_{uc}$. In fact,

$$(\mathcal{OP}'_{uc}) \qquad \begin{array}{lll} \min_x & f(x,p) & \\ & G(x,p) & \in \quad K \end{array} \tag{7.3}$$

is equivalent to the problem:

$$(\mathcal{OP}''_{uc}) \qquad \begin{array}{lll} \min_{t,x} & t & \\ & t - f(x,p) & \geq \quad 0, \\ & G(x,p) & \in \quad K. \end{array} \tag{7.4}$$

This last problem has all its uncertainties in its constraints.

Let us now apply the constraint-robust optimization approach to a multi-period portfolio selection problem:

7.2.1 Robust Multi-Period Portfolio Selection

This part of the notes is adapted from an article by Ben-Tal, Margalit, and Nemirovski [2]. We consider an investor who currently holds the following portfolio: $x^0 = (x_1^0, \ldots, x_n^0)$, where x_i^0 denotes the number of shares of asset i in the portfolio, for $i = 1, \ldots, n$. Also, let x_0^0 denote her cash holdings. She is trying to determine how to adjust her portfolio in the next L investment periods to maximize her total wealth at the end of period L.

We use the following decision variables to model this multi-period portfolio selection problem: b_i^l denotes the number of additional shares of asset i bought at the beginning of period l and s_i^l denotes the number of asset i shares sold at the beginning of period l,

for $i = 1, \ldots, n$ and $l = 1, \ldots, L$. Then, the number of shares of asset i in the portfolio at the beginning of period l, denoted x_i^l, is given by the following simple equation:

$$x_i^l = x_i^{l-1} - s_i^l + b_i^l, \quad i = 1, \ldots, n, \ l = 1, \ldots, L.$$

Let P_i^l denote the price of a share of asset i in period l. For initial prices, without loss of generality we choose $P_i^0 = 1$, for all $i = 0, \ldots, n$; we can always normalize the x^0 quantities if necessary. We make the assumption that the cash account earns no interest so that $P_0^l = 1, \forall l$. This is not a restrictive assumption either–we can always reformulate the problem in this way via a change of numeraire.

We assume that proportional transaction costs are paid on asset purchases and sales and denote them with α_i^l and β_i^l for sales and purchases, respectively, for asset i and period l. We assume that α_i^l's and β_i^l's are all known at the beginning of period 0, although they can vary from period to period and from asset to asset. Transaction costs are paid from the investor's cash account and therefore, we have the following balance equation for the cash account:

$$x_0^l = x_0^{l-1} + \sum_{i=1}^{n}(1-\alpha_i)P_i^l s_i^l - \sum_{i=1}^{n}(1+\beta_i)P_i^l b_i^l, l = 1, \ldots, L.$$

This balance condition indicates that the cash available at the beginning of period l is the sum of last period's cash holdings and the proceeds from sales (discounted by transaction costs) minus the cost of new purchases. For technical reasons, we will replace the equation above with an inequality, effectively allowing the investor "burn" some of her cash if she wishes to:

$$x_0^l \leq x_0^{l-1} + \sum_{i=1}^{n}(1-\alpha_i)P_i^l s_i^l - \sum_{i=1}^{n}(1+\beta_i)P_i^l b_i^l, l = 1, \ldots, L.$$

The objective of the investor, as we mentioned above, is to maximize her total wealth at the end of period L. This objective can be represented as follows:

$$\max \sum_{i=1}^{n} P_i^L x_i^L.$$

If we assume that all the future prices P_i^l are known at the time this investment problem is to be solved, we obtain the following deterministic optimization problem:

$$\begin{array}{lrcl} \max_{x,s,b} \sum_{i=0}^{n} & P_i^L x_i^L & & \\ & x_0^l & \leq & x_0^{l-1} + \sum_{i=1}^{n}(1-\alpha_i)P_i^l s_i^l - \sum_{i=1}^{n}(1+\beta_i)P_i^l b_i^l, l = 1, \ldots, L \\ & x_l^i & = & x_i^{l-1} - s_i^l + b_i^l, \quad i = 1, \ldots, n, \ l = 1, \ldots, L \\ & s_i^l & \geq & 0, \quad i = 1, \ldots, n, \ l = 1, \ldots, L \\ & b_i^l & \geq & 0, \quad i = 1, \ldots, n, \ l = 1, \ldots, L \\ & x_i^l & \geq & 0, i = 0, \ldots, n, \ l = 1, \ldots, L. \end{array} \tag{7.5}$$

This is, in fact, a linear programming problem that can be solved easily using the simplex method or interior-point methods. The nonnegativity constraints on x_i^l's disallow short sales and borrowing–these constraints are not essential to the model and can be removed to allow short sales on a subset of the assets or to allow borrowing. Observe that the investor would, of course, never choose to burn money if she is trying to maximize her final wealth. Therefore, the cash balance inequalities will always be satisfied with equality in an optimal solution of this problem.

In a realistic setting, we do not know P_i^l's in advance and therefore can not solve the optimal portfolio allocation problem as the linear program we developed above. Instead, we will develop a robust optimization model. Since the objective function involves uncertain parameters P_i^L, we first reformulate the problem as in (7.4) to move all the uncertainty to the constraints:

$$\begin{array}{lrcl} \max_{x,s,b,t} & t & & \\ & t & \leq & \sum_{i=0}^{n} P_i^L x_i^L \\ & x_0^l & \leq & x_0^{l-1} + \sum_{i=1}^{n}(1-\alpha_i)P_i^l s_i^l - \sum_{i=1}^{n}(1+\beta_i)P_i^l b_i^l, l = 1,\ldots,L \\ & x_l^i & = & x_i^{l-1} - s_i^l + b_i^l, \quad i = 1,\ldots,n,\ l = 1,\ldots,L \\ & s_i^l & \geq & 0, \quad i = 1,\ldots,n,\ l = 1,\ldots,L \\ & b_i^l & \geq & 0, \quad i = 1,\ldots,n,\ l = 1,\ldots,L \\ & x_i^l & \geq & 0, i = 0,\ldots,n,\ l = 1,\ldots,L. \end{array} \tag{7.6}$$

The first two constraints of this reformulation are the constraints that are affected by uncertainty and we would like to find a solution that satisfies these constraints for most possible realizations of the uncertain parameters P_i^l. To determine the robust version of these constraints, we need to choose an appropriate "uncertainty set" for these uncertain parameters and we follow a 3-sigma approach (as in engineering and statistical applications) for this purpose.

Future prices can be assumed to be random quantities. Let us denote the expected value of the vector $P^l = \begin{bmatrix} P_1^l \\ \vdots \\ P_n^l \end{bmatrix}$ with $\mu^l = \begin{bmatrix} \mu_1^l \\ \vdots \\ \mu_n^l \end{bmatrix}$ and its variance with V^l. First, consider the constraint:

$$t \leq \sum_{i=0}^{n} P_i^L x_i^L.$$

Letting $x^L = (x_1^L, \ldots, x_n^L)$, the expected value and the standard deviation of the right-hand-side expression are given by $(\mu^L)^T x^L = \sum_{i=1}^{n} \mu_i^L x_i^l$ and $\sqrt{(x^L)^T V^L x^L}$ If P_i^L quantities are normally distributed, by requiring

$$t \leq E(RHS) - 3STD(RHS) = (\mu^L)^T x^L - 3\sqrt{(x^L)^T V^L x^L}$$

we would guarantee that the (random) inequality $t \leq \sum_{i=0}^{n} P_i^L x_i^L$ would be satisfied more than 99% of the time, which is equivalent to "always" for an engineer. Therefore, we regard this last inequality as the "robust" version of $t \leq \sum_{i=0}^{n} P_i^L x_i^L$.

We can apply a similar logic to other constraints affected by uncertainty:

$$x_0^l - x_0^{l-1} \leq \sum_{i=1}^{n}(1-\alpha_i)P_i^l s_i^l - \sum_{i=1}^{n}(1+\beta_i)P_i^l b_i^l, l = 1, \ldots, L$$

In this case, the expected value and the variance of the right-hand-side expression are given by the following formulas:

$$\begin{aligned} \mathrm{E}\left[\sum_{i=1}^{n}(1-\alpha_i)P_i^l s_i^l - \sum_{i=1}^{n}(1+\beta_i)P_i^l b_i^l\right] &= (\mu^L)^T D_\alpha^l s^l - (\mu^L)^T D_\beta^l b^l \\ &= (\mu^L)^T \begin{bmatrix} D_\alpha^l & -D_\beta^l \end{bmatrix} \begin{bmatrix} s^l \\ b^l \end{bmatrix}, \end{aligned}$$

and

$$\mathrm{Var}\left[\sum_{i=1}^{n}(1-\alpha_i)P_i^l s_i^l - \sum_{i=1}^{n}(1+\beta_i)P_i^l b_i^l\right] = \begin{bmatrix} s^l & b^l \end{bmatrix} \begin{bmatrix} D_\alpha^l \\ -D_\beta^l \end{bmatrix} V^l \begin{bmatrix} D_\alpha^l & -D_\beta^l \end{bmatrix} \begin{bmatrix} s^l \\ b^l \end{bmatrix}.$$

Above, D_α and D_β are the diagonal matrices

$$D_\alpha := \begin{bmatrix} (1-\alpha_1^l) & & \\ & \ddots & \\ & & (1-\alpha_n^l) \end{bmatrix}, \text{ and } D_\beta := \begin{bmatrix} (1+\beta_1^l) & & \\ & \ddots & \\ & & (1+\beta_n^l) \end{bmatrix},$$

$s^l = (s_1^l, \ldots, s_n^l)$, and $b^l = (b_1^l, \ldots, b_n^l)$. Replacing

$$x_0^l - x_0^{l-1} \leq \sum_{i=1}^{n}(1-\alpha_i)P_i^l s_i^l - \sum_{i=1}^{n}(1+\beta_i)P_i^l b_i^l, l = 1, \ldots, L$$

with

$$x_0^l - x_0^{l-1} \leq (\mu^L)^T \begin{bmatrix} D_\alpha^l & -D_\beta^l \end{bmatrix} \begin{bmatrix} s^l \\ b^l \end{bmatrix} - 3\sqrt{\begin{bmatrix} s^l & b^l \end{bmatrix} \begin{bmatrix} D_\alpha^l \\ -D_\beta^l \end{bmatrix} V^l \begin{bmatrix} D_\alpha^l & -D_\beta^l \end{bmatrix} \begin{bmatrix} s^l \\ b^l \end{bmatrix}}$$

we obtain a "robust" version of the constraint. By satisfying this robust constraint we can guarantee that the original constraint will be satisfied "almost always", no matter what happens to the uncertain parameters.

Our approach above corresponds to choosing the uncertainty sets for the uncertain parameter vectors P^l in the following manner:

$$\mathcal{U}^l := \{P^l : \sqrt{(P^l - \mu^l)^T (V^l)^{-1} (P^l - \mu^l)} \leq 3\}, l = 1, \ldots, L$$

The complete uncertainty set $\mathcal{U}$ for all the uncertain parameters is the Cartesian product of the sets $\mathcal{U}^l$: $\mathcal{U} = \mathcal{U}^1 \times \ldots \times \mathcal{U}^L$.

The resulting problem has nonlinear constraints, because of the square-roots and quadratic terms within the square-roots. Fortunately, however, these constraints can be written as *second order cone* constraints and result in a *second order cone optimization* problem. This is a special and simple case of more general conic optimization problems and can be solved efficiently using interior-point methods.

7.3 Solution Robustness

Another important robustness concept is *solution robustness*. This refers to solutions that will remain close to optimal for all possible realizations of the uncertain problem parameters, and for this reason we prefer the alternative term *objective robust* for such solutions. Since such solutions may be difficult to obtain, especially when uncertainty sets are relatively large, an alternative goal for objective robustness is to find solutions whose worst-case behavior is optimized. Worst-case behavior of a solution corresponds to the value of the objective function for the worst possible realization of the uncertain data for that particular solution. Here is a mathematical model that addresses objective robustness: Consider an optimization problem of the form:

$$(\mathcal{OP}_{uo}) \qquad \begin{array}{ll} \min_x & f(x,p) \\ & x \in S. \end{array} \tag{7.7}$$

Here, S is the (certain) feasible set and f is the objective function that depends on uncertain parameters p. Assume as above that $\mathcal{U}$ is the uncertainty set that contains all possible values of the uncertain parameters p. Then, an objective robust solution can be obtained by solving:

$$(\mathcal{ORCOP}) \qquad \min_{x \in S} \quad \max_{p \in \mathcal{U}} f(x,p). \tag{7.8}$$

Let us now explore some portfolio selection models that incorporate the uncertainty of problem inputs:

7.3.1 Robust Portfolio Selection

This section is adapted from the article [6]. Recall that Markowitz' mean-variance optimization problem can be stated in the following form that combines the reward and risk in the objective function:

$$\max_{x \in \mathcal{X}} \mu^T x - l x^T Q x. \tag{7.9}$$

Here μ_i is an estimate of the expected return of security i, q_{ii} is the variance of this return, q_{ij} is the covariance between the returns of securities i and j, λ is a risk-aversion constant used to trade-off the reward (expected return) and risk (portfolio variance). The set $\mathcal{X}$ is the set of feasible portfolios which may carry information on short-sale restrictions, sector distribution requirements, etc. Since such restrictions are predetermined, we can assume that the set $\mathcal{X}$ is known without any uncertainty at the time the problem is solved.

Recall also that solving the problem above for different values of l one obtains what is known as the *efficient frontier* of the set of feasible portfolios. The optimal portfolio will be different for individuals with different risk-taking tendencies, but it will always be on the efficient frontier.

One of the limitations of this model is its need to accurately estimate the expected returns and covariances. In [1], Bawa, Brown, and Klein argue that using estimates of the unknown expected returns and covariances leads to an *estimation risk* in portfolio choice, and that methods for optimal selection of portfolios must take this risk into account. Furthermore, the optimal solution is sensitive to perturbations in these input parameters—a small change in the estimate of the return or the variance may lead to a large change in the corresponding solution, see, for example, [8, 9]. This attribute is unfavorable since the modeler may want to periodically rebalance the portfolio based on new data and may incur significant transaction costs to do so. Furthermore, using point estimates of the expected return and covariance parameters do not respond to the needs of a conservative investor who does not necessarily trust these estimates and would be more comfortable choosing a portfolio that will perform well under a number of different scenarios. Of course, such an investor cannot expect to get better performance on some of the more likely scenarios, but will have insurance for more extreme cases. All these arguments point to the need of a portfolio optimization formulation that incorporates robustness and tries to find a solution that is relatively insensitive to inaccuracies in the input data. Since all the uncertainty is in the objective function coefficients, we seek a "solution robust" portfolio, as outlined in the introduction to this section.

For *robust portfolio optimization* we consider a model that allows return and covariance matrix information to be given in the form of intervals. For example, this information may take the form "The expected return on security j is between %8 and %10." rather than claiming that it is %9. Mathematically, we will represent this information as membership in the following set:

$$\mathcal{U} = \{(\mu, Q) : \mu^L \leq \mu \leq \mu^U, \quad Q^L \leq Q \leq Q^U, \quad Q \succeq 0\}, \tag{7.10}$$

where μ^L, μ^U, Q^L, Q^U are the extreme values of the intervals we just mentioned. The restriction $Q \succeq 0$ is necessary since Q is a covariance matrix and, therefore, must be positive semidefinite. These intervals may be generated in different ways. An extremely cautious modeler may want to use historical lows and highs of certain input parameters as the range of their values. One may generate different estimates using different scenarios on the general economy and then combine the resulting estimates. Different analysts may produce different estimates for these parameters and one may choose the extreme estimates as the endpoints of the intervals. One may choose a confidence level and then generate estimates of covariance and return parameters in the form of prediction intervals.

We want to find a portfolio that maximizes the objective function in (7.9) in the worst case realization of the input parameters μ and Q from their uncertainty set $\mathcal{U}$ in (7.10). Given these considerations the robust optimization problem given in ($\mathcal{ORQP}$) takes the following form

$$\max_{x \in \mathcal{X}} \{ \min_{(\mu,Q) \in \mathcal{U}} \mu^T x - l x^T Q x \} \tag{7.11}$$

which is equivalent to $\min_{x \in \mathcal{X}} \{\max_{(\mu,Q) \in \mathcal{U}} -\mu^T x + l x^T Q x\}$. This problem can be ex-

pressed as a *saddle-point problem* and be solved using the technique outlined in [6].

7.3.2 Robust Asset Allocation: A Case Study

This material in this section is adapted from the article [16]. We apply the robust optimization approach discussed in the previous section to an asset allocation problem. We consider a universe of 5 asset classes: large cap growth stocks, large cap value stocks, small cap growth stocks, small cap value stocks, and fixed income securities. To represent each asset class, we use a monthly log-return time series of corresponding market indices: Russell 1000 growth and value indices for large cap stocks, Russell 2000 growth and value indices for small cap stocks, and Lehman Brothers US Intermediate Government/Credit Bond index for fixed income securities. Lehman Brothers U.S. Intermediate Government/Credit Bond Index is an unmanaged index generally representative of government and investment-grade corporate securities with maturities of 1-10 years. Our time series data spans the period January 1978 (the inception date of Russell indices) to July 2002, a total of $n = 283$ months.

Using this data, we computed the historical means and covariances of the five indices mentioned above. Further, we determined the uncertainty sets for the actual values of these parameters (lower and upper bound vectors and matrices on means and covariances) using a bootstrapping strategy. Namely, a time series of length n was chosen for each index by bootstrapping from the available observations and means and covariances were computed for these series. This process was repeated 3000 times and the quantiles of the statistics were computed. Table 1 lists the 2.5, 50, and 97.5 percentiles for means of monthly returns and covariances of these returns. The first and last entry for each value was used in the description of the uncertainty set.

$10^{-2}\times$	Ru 1000 Gr	Ru 1000 Va	Ru 2000 Gr	Ru 2000 Va	LB IT Gov/Cre
2.5 percentile	0.3187	0.6351	-0.1480	0.5797	0.5788
50 percentile	0.9621	1.1134	0.7291	1.1757	0.7442
97.5 percentile	1.5946	1.6031	1.5568	1.7486	0.9099

$10^{-3}\times$	Ru 1000 Gr	Ru 1000 Va	Ru 2000 Gr	Ru 2000 Va	LB IT Gov/Cre
Ru 1000 Gr	2.2283 2.8771 3.6662				
Ru 1000 Va	1.3628 1.8353 2.4660	1.3057 1.7379 2.2796			
Ru 2000 Gr	2.4206 3.2729 4.3474	1.4273 2.1309 3.0909	3.8896 5.1327 6.7269		
Ru 2000 Va	1.3054 1.9130 2.7692	1.1325 1.6841 2.4485	2.1489 3.0751 4.3566	1.6569 2.4125 3.5011	
LB IT Gov/Cre	0.0529 0.1370 0.2221	0.0712 0.1466 0.2279	-0.0277 0.0896 0.2048	0.0234 0.1184 0.2181	0.1350 0.1854 0.2502

Table 7.1: 2.5, 50, and 97.5 percentiles of mean monthly log-returns as well as the entries of the covariance matrix obtained from bootstrapped samples. Only the lower diagonal entries in the covariance matrix are listed for brevity.

Using the data presented above, we generated the standard and robust efficient frontiers. Figure 7.1 depicts the standard efficient frontier obtained by using the 50 percentile values for expected returns and covariances as inputs and the composition of the portfolios on the efficient frontier. Lowest risk efficient portfolios are obtained, as expected, using the fixed income securities. As one moves along the efficient frontier toward the efficient portfolio with the highest expected return, fixed income securities are gradually replaced by a mixture of large-cap and small-cap value stocks. Close to the high-return end of the frontier, large-cap stocks are also phased out and one gets a portfolio consisting entirely of small cap value stocks.

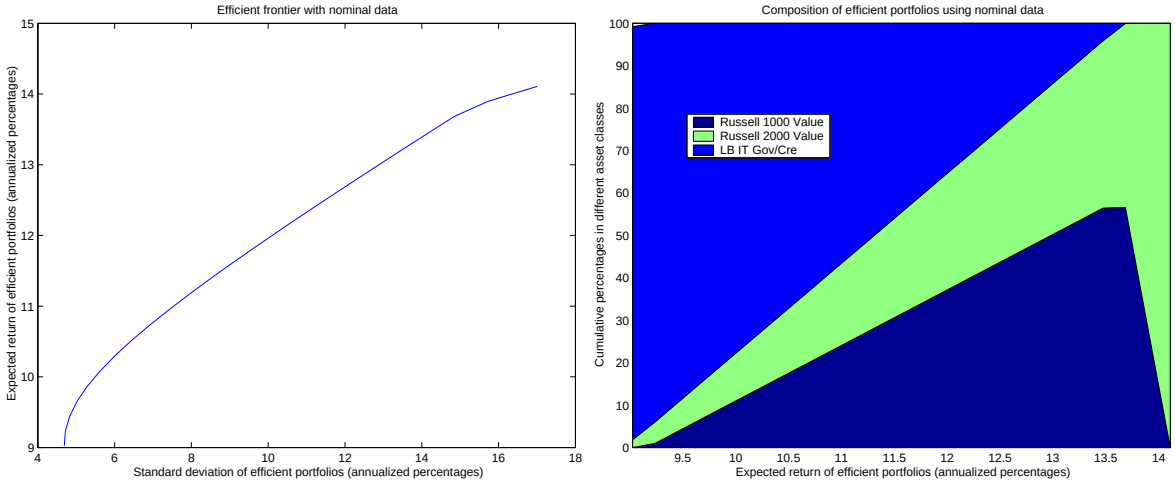


Figure 7.1: The efficient frontier and the composition of the efficient portfolios found using the classical MVO approach without any consideration of input uncertainty.

Figure 7.2 shows the corresponding graphs for robust efficient portfolios. To find the robust efficient portfolios, we used the 2.5 and 97.5 percentiles listed in Table 1 as the lower and upper bounds μ^L, μ^U, Q^L, and Q^U. We observe that Q^U obtained in this manner is a positive definite matrix. In this case, the the robust efficient portfolios can be found using the classical mean-variance optimization approach with inputs μ^L and Q^U and the (more involved) saddle-point approach outlined in the previous section is not necessary. Note that the figures for the robust efficient portfolios depict the *worst-case* values of the expected returns and standard deviations, so returns are substantially smaller and volatilities are higher. In contrast to the classical efficient portfolios, robust efficient portfolios never utilize small cap value stocks and instead concentrate all equity holdings in large cap value stocks (with the exception of a small allocation to large cap growth stocks at the low-risk end of the frontier).

Finally, we compare the (σ, μ) frontiers for classical and robust efficient portfolios. First, we plot the standard deviation-expected return profiles of the generated portfolios assuming that the expected returns and covariances were actually equal to the point estimates used for the classical mean-variance optimization approach. Under

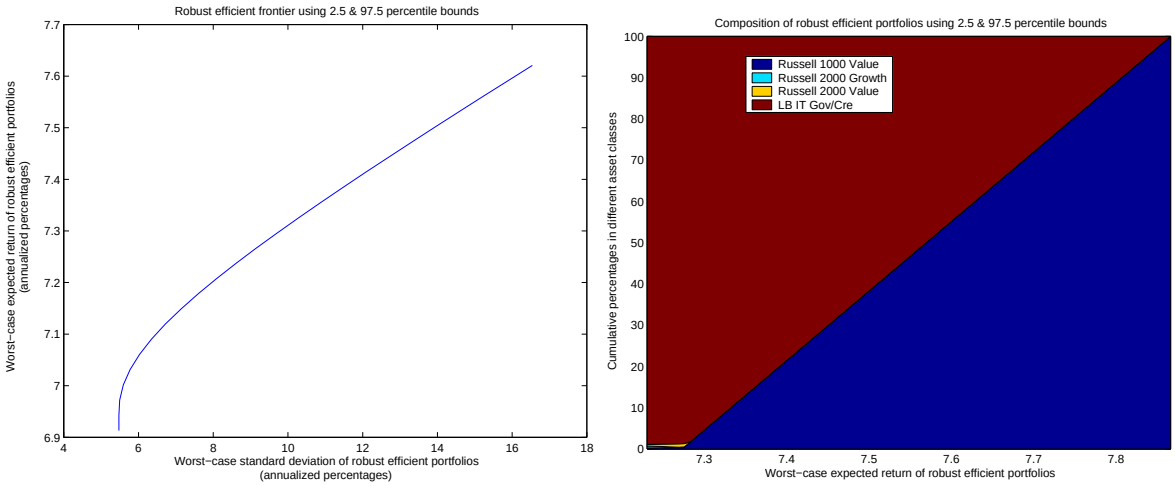


Figure 7.2: The efficient frontier and the composition of the efficient portfolios found using the robust asset allocation approach. 2.5 and 97.5 percentiles of means and covariances of bootstrapped samples were used to describe the uncertainty intervals for these inputs.

this scenario, portfolios coming from the classical MVO approach only slightly outperform the portfolios generated with the worst-case in mind. Next, we plot the standard deviation-expected return profiles of the generated portfolios assuming that actual expected returns and covariances were the worst-case values within the lower and upper bounds used for robust optimization. Figure 7.3 shows these two graphs. Compared to the nominal case, the difference in the worst-case performances of the two sets of efficient portfolios is dramatically different. Namely, the performance of classical efficient portfolios deteriorate significantly at the high-return end with worst-case inputs.

We conclude this section by noting that robust portfolio optimization approaches can also be implemented in the framework of *factor models*, i.e., when the interdependencies of stock returns are explained through a small number of factors. In [5], Goldfarb and Iyengar investigate such problems and show that in this case, the robust portfolio selection problem reduces to a second-order cone programming problem when the uncertainty sets are ellipsoids. Second-order cone problems can be solved efficiently using interior-point approaches similar to the one presented in the previous section.

7.4 Exercises

1. Recall that we considered the following *two-stage stochastic linear program with recourse* in Section 6.2.

$$\begin{array}{rrcrcl} \max & (c^1)^T x^1 & + & E[\max c^2(\omega)^T x^2(\omega)] & & \\ & A^1 x^1 & & & = & b^1 \\ & B^2(\omega) x^1 & + & A^2(\omega) x^2(\omega) & = & b^2(\omega) \\ & x^1 \geq 0, & & x^2(\omega) \geq 0. & & \end{array} \tag{7.12}$$

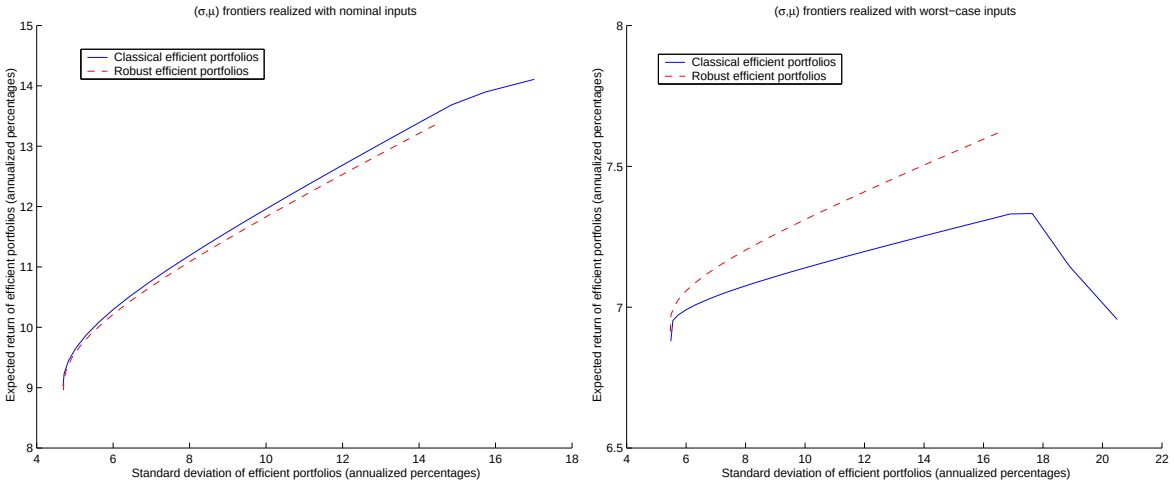


Figure 7.3: (σ, μ)-profiles of classical and robust efficient portfolios when actual moments are (i) equal to their point estimates, (ii) equal to their worst possible values within given bounds.

In this problem, it was assumed the uncertainty in ω was of "random" nature, and therefore, the stochastic programming approach was appropriate. Now consider the case where ω is not a random variable but is known to belong to an uncertainty set $\mathcal{U}$. Formulate a *two-stage robust linear program with recourse* using the ideas developed in Section 8. Next, assume that B^2 and A^2 are certain (they do not depend on ω), but b^2 and c^2 are uncertain and depend affinely on ω: $b^2(\omega) = b^2 + P\omega$ and $c^2(\omega) = c^2 + R\omega$, where b^2, c^2, P, R are (certain) vectors/matrices of appropriate dimension. Also, assume that $\mathcal{U} = \{\omega : \sum_i d_i w_i^2 \leq 1\}$ for some positive constants d_i. Can you simplify the two-stage robust linear program with recourse under these assumptions?

2. When we studied model robustness on the multi-period portfolio selection problem, we replaced the constraint

$$t \leq \sum_{i=0}^{n} P_i^L x_i^L$$

(which has a random right-hand-side) with the following "robust" constraint:

$$t \leq E(RHS) - 3STD(RHS) = (\mu^L)^T x^L - 3\sqrt{(x^L)^T V^L x^L},$$

where μ^L and V^L denote the expected value vector and the (positive definite) covariance matrix of the random vector P_i^L.

Given μ^L and V^L, consider the following uncertainty set for the uncertain parameters P_i^L:

$$\mathcal{U}^L := \{P^L : \sqrt{(P^L - \mu^L)^T (V^L)^{-1} (P^L - \mu^L)} \leq 3\}.$$

Show that

$$t \leq \sum_{i=0}^{n} P_i^L x_i^L, \quad \forall P^L \in \mathcal{U}^L$$

if and only if

$$t \leq (\mu^L)^T x^L - 3\sqrt{(x^L)^T V^L x^L}.$$

Thus, our 3-σ approach is equivalent to the robust formulation of this constraint using an appropriate uncertainty set.

(Hint: You may first want to show that

$$\mathcal{U}^L = \{\mu^L + (V^L)^{1/2} u : \|u\| \leq 3\}.)$$

3. In Section 7.3.1 we described the robust portfolio selection problem formulated as:

$$\max_{x \in \mathcal{X}} \{ \min_{(\mu,Q) \in \mathcal{U}} \mu^T x - l x^T Q x \} \tag{7.13}$$

where the uncertainty set $\mathcal{U}$ is described as follows:

$$\mathcal{U} = \{(\mu, Q) : \mu^L \leq \mu \leq \mu^U, \quad Q^L \leq Q \leq Q^U, \quad Q \succeq 0\}.$$

Now we consider a special case of this problem where we make the following assumptions

- $x \geq 0, \ \forall x \in \mathcal{X}$ (i.e., $\mathcal{X}$ includes nonnegativity constraints)
- Q^U is positive semidefinite.

Under these assumptions, show that (7.13) reduces to the following maximization problem:

$$\max_{x \in \mathcal{X}} \left(\mu^L\right)^T x - l x^T Q^U x. \tag{7.14}$$

Observe that this new problem is a simple concave quadratic maximization problem and can be solved easily using, for example, interior-point methods. (Hint: Note that the objective function of (7.13) is separable in μ and Q and that $x^T Q x = \sum_{i,j} q_{ij} x_{ij}$ with $x_{ij} = x_i x_j \geq 0$ when $x \geq 0$.)

4. For a given constant l, expected return vector μ, and a positive definite covariance matrix Q consider the following MVO problem:

$$\max_{x \in \mathcal{X}} \mu^T x - l x^T Q x, \tag{7.15}$$

where $\mathcal{X} = \{x : e^T x = 1\}$ with $e = [1 \ 1 \ \ldots \ 1]^T$. Let $z(\mu, Q)$ represent the optimal value of this problem. Determine $z(\mu, Q)$ as an explicit function of μ and Q. Next, assume that μ and Q are uncertain and belong to the uncertainty set $\mathcal{U} := \{(\mu_i, Q_i) : i = 1, \ldots, m\}$, i.e., we have a finite number of scenarios for μ

and Q. Assume also that $z(\mu_i, Q_i) > 0 \ \forall i$. Now formulate the following robust optimization problem: Find a feasible portfolio vector x such that the objective value with this portfolio under each scenario is within 10% of the optimal objective value corresponding to that scenario. Discuss how this problem can be solved. What would be a good objective function for this problem?

Chapter 8

Conic Optimization

Recall the definition of a standard form conic optimization problem from the first chapter:

$$(\mathcal{CO}) \qquad \begin{array}{rcl} \min_x \quad c^T x & & \\ Ax & = & b \\ x \in C. & & \end{array} \tag{8.1}$$

Here, C denotes a closed convex cone (see the Appendix for a brief discussion on cones) in a finite-dimensional vector space X. In other words, conic optimization refers to the problem of of minimizing a linear function over the intersection of a translate of a subspace (the region defined by the linear equations $Ax = b$) and a closed convex cone. When $X = \Re^n$ and $C = \Re^n_+$, this problem is the standard form LP. However, this setting is much more general than linear programming since we can use non-polyhedral cones C in the description of these problems.

Conic optimization offers a convenient setting where the sophisticated interior-point algorithms for linear programming problems can be generalized and used very efficiently to solve a large class of convex optimization problems. An advanced discussion on this subject can be found in [10]. Two important subclasses of conic optimization problems are the following:

1. **Second-order cone programming:** This corresponds to the case where C is the second-order cone (also known as the quadratic cone, Lorenz cone, and the ice-cream cone):

$$C_q := \{x = (x_0, x_1, \ldots, x_n) \in \Re^{n+1} : x_0 \geq \|(x_1, \ldots, x_n)\|\}. \tag{8.2}$$

2. **Semidefinite programming:** This corresponds to the case where C is the cone of positive semidefinite matrices of a fixed dimension (say n):

$$C_s^n := \left\{ X = \begin{bmatrix} x_{11} & \cdots & x_{1n} \\ \vdots & \ddots & \vdots \\ x_{n1} & \cdots & x_{nn} \end{bmatrix} \in \Re^{n \times n} : X = X^T, \ X \text{ is positive semidefinite} \right\}. \tag{8.3}$$

During the past decade, there was an intense theoretical and algorithmic study of conic optimization problems that also produced a number of increasingly sophisticated software for several problem classes. Interested readers can obtain additional information on such software by following the software link of the following page dedicated to semidefinite programming:

`http://www-user.tu-chemnitz.de/~helmberg/semidef.html`

8.1 Conic Optimization Models and Tools in Finance

We have already seen examples of conic optimization models in our discussion of robust optimization. Specifically, our discussion on "Robust Multi-Period Portfolio Selection" in Section 7.2.1 led to an optimization problem with constraints of the form

$$t \leq (\mu^L)^T x^L - 3\sqrt{(x^L)^T V^L x^L}$$

which, we argued, can be represented as second-order cone constraints. Similarly, in Section 7.3.1, when we discussed the robust portfolio selection models, we considered uncertainty sets that used the positive semi-definiteness restriction in their definition. To optimize over such sets, one has to solve conic optimization problems as well. Here we discuss two additional examples from finance that lead to conic optimization problems, further emphasizing the importance of these sophisticated models for modern financial mathematics.

8.1.1 Minimum Risk Arbitrage

The material in this subsection is based on the article [11]. Consider an investment environment with n financial securities whose future price vector $r \in \Re^n$ is a random variable. Let $p \in \Re^n$ represent the current prices of these securities. If the investor chooses a portfolio $x = (x_1, \ldots, x_n)$ that satisfies

$$p^T x < 0$$

and the realization $\tilde{r}$ at the end of the investment period of the random variable r satisfies

$$\tilde{r}^T x \geq 0$$

then the investor would make money: S/he forms a portfolio with negative cash flow (pocketing money) and the portfolio has a nonnegative value at the end. If the investor can choose a portfolio x such that $p^T x < 0$ and

$$\text{Prob}[r^T x \geq 0] = 1$$

then, there is an arbitrage opportunity (type A).

Since arbitrage opportunities generally do not exist (or at least, do not exist for long periods), one might be interested in the alternative notion of "minimum risk arbitrage". This concept is developed using a similar construction to what we have seen in Section 7.2.1. Let μ and Q represent the expected future price vector and covariance matrix of the random vector r. Then, as in Section 7.2.1, the random inequality

$$r^T x \geq 0$$

can be replaced by the following deterministic approximation:

$$\mu^T x - \theta\sqrt{x^T Q x} \geq 0.$$

As in Section 7.2.1, choosing $\theta = 3$ would correspond to the 3-σ approach of engineering. When returns are normally distributed, satisfying this last inequality with $\theta = 3$ would ensure that

$$\text{Prob}[r^T x \geq 0] \geq 0.99.$$

Therefore, if we find an x satisfying

$$\mu^T x - \theta\sqrt{x^T Q x} \geq 0, p^T x < 0$$

for a large enough positive value of θ we have an approximation of an arbitrage opportunity. Note that, by relaxing the constraint $p^T x < 0$ as $p^T x \leq 0$ or as $p^T x \leq -\varepsilon$, we obtain a conic feasibility system. Therefore, the resulting system can be solved using the conic optimization approaches.

8.1.2 Approximating Covariance Matrices

Given a vector of random variables, the covariance matrix of these random variables is one of the most important entities describing the joint behavior of these random variables. Covariance matrices are encountered frequently is financial mathematics, for example, in mean-variance optimization, in forecasting, in time-series modeling, etc. Often, true values of covariance matrices are not observable and one must use estimates. Since estimation is often performed entry-by-entry, the resulting estimate of the covariance matrix may be "improper" in the sense that it may not satisfy the following basic property of any covariance matrix: positive semi-definiteness.

Lack of positive semi-definiteness in a covariance matrix estimate is a serious problem–one might be led to think that there are portfolios with negative variances! Assume then, that we have an estimate $\hat{Q} \in \Sigma^n$ of a covariance matrix and that $\hat{Q}$ is not positive semidefinite. Here, S^n denotes the space of symmetric $n \times n$ matrices. An important question in this scenario is the following: What is the "closest" positive

semidefinite matrix to $\hat{Q}$? Here, the definition of "close" may be subjective–among a few alternatives we consider the following popular measure:

$$d_F(Q, \hat{Q}) = \sqrt{\sum_{i,j}(Q_{ij} - \hat{Q}_{ij})^2}.$$

Above, F stands for Frobenius, as the measure d_F is nothing but the Frobenius norm of $(Q - \hat{Q})$.

Therefore, we have the following optimization problem: Given $\hat{Q} \in S^n$,

$$\begin{gathered} \min d_F(Q, \hat{Q}) \\ Q \in C_s^n \end{gathered}$$

where C_s^n is as defined in (8.3). Furthermore, introducing a dummy variable t, we can rewrite the last problem above as:

$$\begin{gathered} \min t \\ d_F(Q, \hat{Q}) \leq t \\ Q \in C_s^n. \end{gathered}$$

It is easy to see that the inequality $d_F(Q, \hat{Q}) \leq t$ can be written as a second-order cone constraint, and therefore, the formulation above can be transformed into a conic optimization problem.

8.2 Exercises

1. A vector $(y^0, y^1) \in \Re \times \Re^k$ belongs to the $k+1$ dimensional second-order cone (also known as the quadratic cone, Lorentz cone, ice-cream cone) if it satisfies the following inequality:

   $$y^0 \geq \|y^1\|_2.$$

   Constraints of the form above are called second-order cone constraints. Show that the constraint

   $$t \leq (\mu^L)^T x^L - 3\sqrt{(x^L)^T V^L x^L}$$

   can be represented as a second-order cone constraint using an appropriate change of variables. You can assume that V^L is a given positive definite matrix.

Appendix A

Convexity

Convexity is an important concept in mathematics, and especially in optimization, that is used to describe certain sets and certain functions. Convex sets and convex functions are related but separate mathematical entities.

Let x and y be given points in some vector space. Then, for any $l \in [0,1]$, the point $lx + (1-l)y$ is called a *convex combination* of x and y. The set of all convex combinations of x and y is the line segment joining these two points.

A subset S of a given vector space X is called a *convex set* if $x \in S$, $y \in S$, and $\lambda \in [0,1]$ always imply that $\lambda x + (1-\lambda)y \in S$. In other words, a convex set is characterized by the following property: for any two given points in the set, the line segment connecting these two points lies entirely in the set.

Polyhedral sets are sets defined by linear equalities and inequalities. So, for example, the feasible region of a linear optimization problem is a polyhedral set. It is a straightforward exercise to show that polyhedral sets are convex.

Given a convex set S, a function $f : S \to \Re$ is called a *convex function* if $\forall x \in S, y \in S$ and $\lambda \in [0,1]$ the following inequality holds:

$$f(\lambda x + (1-\lambda)y) \leq \lambda f(x) + (1-\lambda)f(y).$$

We say that f is a *strictly convex function* if $x \in S, y \in S$ and $\lambda \in (0,1)$ implies the following strict inequality:

$$f(\lambda x + (1-\lambda)y) < \lambda f(x) + (1-\lambda)f(y).$$

A function f is *concave* if $-f$ is convex. Equivalently, f is concave, if $\forall x \in S, y \in S$ and $\lambda \in [0,1]$ the following inequality holds:

$$f(\lambda x + (1-\lambda)y) \geq \lambda f(x) + (1-\lambda)f(y).$$

A function f is *strictly concave* if $-f$ is strictly convex.

Given $f : S \to \Re$ with $S \subset X$, epi(f)–the epigraph of f, is the following subset of $X \times \Re$:

$$\text{epi}(f) := \{(x,r) : x \in S, f(x) \leq r\}.$$

f is a convex *function* if and only if epi(f) is a convex *set*.

For a twice-continuously differentiable function $f : S \to \Re$ with $S \subset \Re$, we have a simple characterization of convexity: f is convex on S if and only if $f''(x) \geq 0,\ \forall x \in S$. For multivariate functions, we have the following generalization: If $f : S \to \Re$ with $S \subset \Re^n$ is twice-continuously differentiable, then f is convex on S if and only if $\nabla^2 f(x)$ is positive semidefinite for all $x \in S$. Here, $\nabla^2 f(x)$ denotes the (symmetric) Hessian matrix of f; namely, $[\nabla^2 f(x)]_{ij} = \frac{\partial^2 f(x)}{\partial x_i \partial x_j}, \forall i, j$. Recall that a symmetric matrix $H \in \Re^{n \times n}$ is positive semidefinite (positive definite) if $y^T H y \geq 0,\ \forall y \in \Re^n$ ($y^T H y > 0,\ \forall\ y \in \Re^n, y \neq 0$).

The following theorem is one of the many reasons for the importance of convex functions and convex sets for optimization:

Theorem A.1 *Consider the following optimization problem:*

$$(\mathcal{OP}) \quad \begin{array}{ll} \min_x & f(x) \\ s.t. & x \in S \end{array} \tag{A.1}$$

If S is a convex set and if f is a convex function of x on S, then all local optimal solutions of $\mathcal{OP}$ are also global optimal solutions.

Appendix B

Cones

A *cone* is a set that is closed under positive scalar multiplication. In other words, a set C is a cone if $lx \in C$ for all $l \geq 0$ and $x \in C$. A cone is called *pointed* if it does not include any lines. We will generally be dealing with closed, convex, and pointed cones. Here are a few important examples:

- $C_l := \{x \in \Re^n : x \geq 0\}$, the non-negative orthant. In general, any set of the form $C := \{x \in \Re^n : Ax \geq 0\}$ for some matrix $A \in \Re^{m \times n}$ is called a *polyhedral cone*. The subscript l is used to indicate that this cone is defined by linear inequalities.
- $C_q := \{x = (x_0, x_1, \ldots, x_n) \in \Re^{n+1} : x_0 \geq \|(x_1, \ldots, x_n)\|\}$, the second-order cone. This cone is also called the quadratic cone (hence the subscript q), Lorentz cone, and the ice-cream cone.
- $C_s := \left\{ X = \begin{bmatrix} x_{11} & \cdots & x_{1n} \\ \vdots & \ddots & \vdots \\ x_{n1} & \cdots & x_{nn} \end{bmatrix} \in \Re^{n \times n} : X = X^T,\ X \text{ is positive semidefinite} \right\}$, the cone of symmetric positive semidefinite matrices.

If C is a cone in a vector space X with an inner product denoted by $\langle \cdot, \cdot \rangle$, then its *dual cone* is defined as follows:

$$C^* := \{x \in X : \langle x, y \rangle \geq 0, \forall y \in C\}.$$

It is easy to see that the nonnegative orthant in $\Re^n$ (with the usual inner product) is equal to its dual cone. The same holds for the second-order cone and the cone of symmetric positive semidefinite matrices, but not for general cones.

The *polar cone* is the negative of the dual cone, i.e.,

$$C^P := \{x \in X : \langle x, y \rangle \leq 0, \forall y \in C\}.$$

Appendix C

A Probability Primer

One of the most basic concepts in probability theory is a *random experiment*, which is an experiment whose outcome can not be determined in advance. In most cases, however, one has a (possibly infinite) set of all possible outcomes of the event; we call this set the *sample space* of the random experiment. For example, flipping a coin is a random experiment, so is the score of the next soccer game between Japan and Korea. The set $\Omega = \{\texttt{heads, tails}\}$ is the sample space of the first experiment, $\Omega = I\!N \times I\!N$ with $I\!N = \{0, 1, 2, \ldots\}$ is the sample space for the second experiment.

Another important concept is an *event*: a subset of the sample space. It is customary to say that an event *occurs* if the outcome of the experiment is in the corresponding subset. So, "Japan beats Korea" is an event for the second experiment of the previous paragraph. A class $\mathcal{F}$ of subsets of a sample space Ω is called a *field* if it satisfies the following conditions:

i) $\Omega \in \mathcal{F}$,

ii) $A \in \mathcal{F}$ implies that $A^c \in \mathcal{F}$, where A^c is the complement of A,

iii) A, $B \in \mathcal{F}$ implies $A \cup B \in \mathcal{F}$.

The second and third conditions are known as *closure under complements and (finite) unions*. If, in addition, $\mathcal{F}$ satisfies

iv) $A_1, A_2, \ldots \in \mathcal{F}$ implies $\cup_{i=1}^{\infty} A_i \in \mathcal{F}$,

then $\mathcal{F}$ is called a σ-field. The condition (iv) is *closure under countable unions*. Note that, for subtle reasons, Condition (iii) does not necessarily imply Condition (iv).

A *probability measure* or *distribution* P is a real-valued function defined on a field $\mathcal{F}$ (whose elements are subsets of the sample space Ω), and satisfies the following conditions

i) $0 \leq P(A) \leq 1$, for $\forall A \in \mathcal{F}$,

ii) $P(\emptyset) = 0$, and $P(\Omega) = 1$,

iii) If $A_1, A_2, \ldots$ is a sequence of disjoint sets in $\mathcal{F}$ and if $\cup_{i=1}^{\infty} A_i \in \mathcal{F}$, then

$$P(\cup_{i=1}^{\infty} A_i) = \sum_{i=1}^{\infty} P(A_i).$$

The last condition above is called *countable additivity*.

A probability measure is said to be *discrete* if Ω has countably many (and possibly finite) number of elements. A *density function* f is a nonnegative valued integrable function that satisfies

$$\int_{\Omega} f(x)dx = 1.$$

A continuous probability distribution is a probability defined by the following relation:

$$P[X \in A] = \int_A f(x)dx,$$

for a density function f.

The collection Ω, $\mathcal{F}$ (a σ-field in Ω), and P (a probability measure on $\mathcal{F}$) is called a *probability space*.

Now we are ready to define a *random variable*. A random variable X is a real-valued function defined on the set Ω [1]. Continuing with the soccer example, the difference between the goals scored by the two teams is a random variable, and so is the "winner", a function which is equal to, say, 1 if the number of goals scored by Japan is higher, 2 if the number of goals scored by Korea is higher, and 0 if they are equal. A random variable is said to be discrete (respectively, continuous) if the underlying probability space is discrete (respectively, continuous).

The *probability distribution* of a random variable X is, by definition, the probability measure P_X in the probability space $(\Omega, \mathcal{F}, P)$:

$$P_X(B) = P[X \in B].$$

The *distribution function* F of the random variable X is defined as:

$$F(x) = P[X \leq x] = P\left[X \in (-\infty, x]\right].$$

For a continuous random variable X with the density function f,

$$F(x) = \int_{-\infty}^{x} f(x)dx$$

[1] Technically speaking, for X to be a random variable, it has to satisfy the condition that for each $B \in \mathcal{B}$, the Euclidean Borel field on $\Re$, the set $\{\omega : X(\omega) \in B\} =: X^{-1}(B) \in \mathcal{F}$. This is a purely technical requirement which is met for discrete probability spaces (Ω is finite or countably infinite) and by any function that we will be interested in.

and therefore $f(x) = \frac{d}{dx}F(x)$.

A *random vector* $\mathbf{X} = (X_1, X_2, \ldots, X_k)$ is a k-tuple of random variables, or equivalently, a function from Ω to $\Re^k$ that satisfies a technical condition similar to the one mentioned in the footnote. The *joint distribution function* F of random variables $X_1, \ldots, X_k$ is defined by

$$F(x_1, \ldots, x_k) = P_{\mathbf{X}}[X_1 \leq x_1, \ldots, X_k \leq x_k].$$

In the special case of $k = 2$ we have

$$F(x_1, x_2) = P_{\mathbf{X}}[X_1 \leq x_1, X_2 \leq x_2].$$

Given the joint distribution function of random variables X_1 and X_2, their *marginal distribution functions* are given by the following formulas:

$$F_{X_1}(x_1) = \lim_{x_2 \to \infty} F(x_1, x_2)$$

and

$$F_{X_2}(x_2) = \lim_{x_1 \to \infty} F(x_1, x_2).$$

We say that random variables X_1 and X_2 are *independent* if

$$F(x_1, x_2) = F_{X_1}(x_1) F_{X_2}(x_2)$$

for every x_1 and x_2.

The *expected value* (*expectation, mean*) of the random variable X is defined by

$$\begin{aligned} E[X] &= \int_\Omega x dF(x) \\ &= \begin{cases} \sum_{x \in \Omega} x P[X = x] & \text{if } X \text{ is discrete} \\ \int_\Omega x f(x) dx & \text{if } X \text{ is continuous} \end{cases} \end{aligned}$$

(provided that the integrals exist) and is denoted by $E[X]$. For a function $g(X)$ of a random variable, the expected value of $g(X)$ (which is itself a random variable) is given by

$$E[g(X)] = \int_\Omega x dF_g(x) = \int_\Omega g(x) dF(x).$$

The *variance* of a random variable X is defined by

$$\begin{aligned} Var[X] &= E\left[(X - E[X])^2\right] \\ &= E[X^2] - (E[X])^2. \end{aligned}$$

The *standard deviation* of a random variable is the square-root of its variance.

For two jointly distributed random variables X_1 and X_2, their *covariance* is defined to be

$$\begin{aligned} Cov(X_1, X_2) &= E\left[(X_1 - E[X_1])(X_2 - E[X_2])\right] \\ &= E[X_1 X_2] - E[X_1]E[X_2] \end{aligned}$$

The *correlation coefficient* of two random variables is the ratio of their covariance to the product of their standard deviations.

For a collection of random variables $X_1, \ldots, X_n$, the expected value of the sum of these random variables is equal to the sum of their expected values:

$$E\left[\sum_{i=1}^{n} X_i\right] = \sum_{i=1}^{n} E[X_i].$$

The formula for the variance of the sum of the random variables $X_1, \ldots, X_n$ is a bit more complicated:

$$Var\left[\sum_{i=1}^{n} X_i\right] = \sum_{i=1}^{n} Var[X_i] + 2 \sum_{1 \le i < j \le n} Cov(X_i, X_j).$$

Appendix D

Newton's Method

To solve optimization problems (such as LPs and QPs) one usually follows the following strategy: First, a set of conditions that has to be satisfied by an optimal solution is determined. Such conditions are called *optimality conditions* for the given problem. These may be linear or nonlinear, equality or inequality constraints[1]. Once the optimality conditions are determined, one looks for solutions for this system. When optimality conditions involve inequality constraints, they become much harder to deal with. Here, we focus on the case where optimality conditions are given by a system of equations.

Suppose that the set of optimality conditions for a given problem have the following form:

$$\begin{array}{rcl} f_1(x_1, x_2, \ldots, x_n) & = & 0 \\ f_2(x_1, x_2, \ldots, x_n) & = & 0 \\ \ddots & & \vdots \\ f_n(x_1, x_2, \ldots, x_n) & = & 0 \end{array} \tag{D.1}$$

Solving this system is a relatively easy task when all the functions f_1 through f_n are linear, for example, by using Gaussian elimination. Unfortunately, this task gets much harder when some or all of the functions $f_1, \ldots, f_n$ are nonlinear. Except for a few special cases, solution of nonlinear systems of equations is often done using an *iterative* approach, where each iteration consists of the solution of a *linear* system. In each iteration, we get a new estimate of the solution, which is obtained by improving the previous estimate in one way or another. Newton's method is the basis of most modern methods for the solution of nonlinear systems of equations and here we describe it briefly.

Let us first consider the one-dimensional case, i.e., when we have only one equation

[1]Recall from calculus that to find the minimum value of a (convex) function, all you did was to find a value where the derivative was zero. So, for such problems 'The derivative equals zero' was the optimality condition.

in one variable. So, for a given nonlinear function f we want to find an x such that

$$f(x) = 0.$$

Assume that f is continuously differentiable and we currently have an estimate x_k for the solution (we will use subscripts for iteration indices in the following discussion). If we write out the first order Taylor series approximation to the function f around x_k we get:

$$f(x_k + \delta) \approx g(\delta) := f(x_k) + \delta f'(x_k).$$

This is equivalent to saying that we can approximate the function f by the line $g(\delta)$ that is tangent to it at x_k. Remember that we want an x such that $f(x) = 0$. If the first order approximation $g(\delta)$ were perfectly good, and if $f'(x_k) \neq 0$ the value of δ that satisfies

$$g(\delta) = f(x_k) + \delta f'(x_k) = 0$$

would give us the move that we need to make from x_k to get to a solution. This value is

$$\delta = -\frac{f(x_k)}{f'(x_k)}$$

and we choose

$$x_{k+1} = x_k + \delta = x_k - \frac{f(x_k)}{f'(x_k)}$$

as our next estimate of the solution to the nonlinear equation $f(x) = 0$. This is the formula for Newton's method and this operation is called the Newton update. We do this repeatedly until $f(x_k) = 0$, or in most cases, until $f(x_k)$ is reasonably small, say, less than some prespecified $\varepsilon > 0$.

We can give a geometric explanation of the procedure we just described: We first find the line tangent to the function at the current iterate, then calculate the point where this line intersects the x-axis, and set the next iterate to this value. See Figure D.1 for an illustration.

Example D.1 *As an example we consider the one dimensional problem*

$$f(x) = x^4 - 4x^3 - 7x^2 + 34x - 24 = 0.$$

Then, the derivative of f is

$$f'(x) = 4x^3 - 12x^2 - 14x + 34$$

and the formula for Newton's method is:

$$x_{k+1} = x_k - \frac{x^4 - 4x^3 - 7x^2 + 34x - 24}{4x^3 - 12x^2 - 14x + 34}.$$

We need to start the method with an initial guess, let us choose $x_0 = 0$. *Then*

$$\begin{aligned} x_1 &= x_0 - \frac{x_0^4 - 4x_0^3 - 7x_0^2 + 34x_0 - 24}{4x_0^3 - 12x_0^2 - 14x_0 + 34} \\ &= 0 - \frac{-24}{34} = \frac{12}{17} = 0.705882352941 \end{aligned}$$

We mentioned above that the next iterate of Newton's method is found by calculating the point where the line tangent to f *at the current iterate intersects the axis. This observation is illustrated in Figure D.1.*

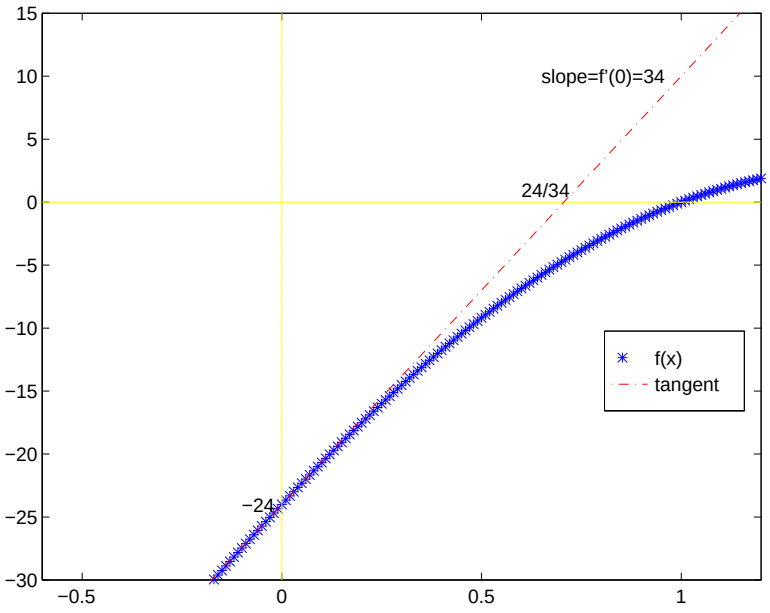


Figure D.1: First step of Newton's method

Since $f(x_1) = f(\frac{12}{17}) = -4.646496090804$ *is not close to zero we continue by substituting* x_1 *into the formula to obtain* $x_2 = 0.943612085452$. *The complete iteration sequence is given in Table D.1.*

A few comments on the speed and reliability of Newton's method are in order. Under favorable conditions, Newton's method converges very fast to a solution of a

Table D.1: Newton's method for Example D.1

k	x_k	$f(x_k)$
0	0	-24.000000000000
1	0.705882352941	-4.646496090804
2	0.943612085452	-0.717979624527
3	0.996932543663	-0.036931796699
4	0.999989873926	-0.000121514224
5	0.999999999889	-0.000000001333
6	1.000000000000	-0.000000000000
7	1.000000000000	0.000000000000

nonlinear equation. Indeed, if x_k is sufficiently close to a solution x_*, then the following relation holds:

$$x_{k+1} - x_* \approx C(x_k - x_*)^2 \text{ with } C = \frac{f''(x_*)}{2f'(x_*)} \tag{D.2}$$

(D.2) indicates that, the error in our approximation $(x_k - x_*)$ is approximately squared in each iteration. This behavior is called the *quadratic convergence* of Newton's method. You can observe that the correct digits are doubled in each iteration of the example above.

However, when the 'favorable conditions' we mentioned above are not satisfied, Newton's method may (and very often does) fail to converge to a solution. Therefore, it often has to be modified before being applied to general problems. Common modifications to Newton's method lead to *line-search methods* and *trust-region methods*; you can learn more about such methods in a nonlinear optimization course.

Next, we will look at the case where there are several equations involving several variables as in (D.1). Let us represent this system as

$$F(x) = 0,$$

where x is a vector of n variables, and $F(x)$ is $\Re^n$-valued function with components $f_1(x), \ldots, f_n(x)$. We repeat the procedure: First, we write the first order Taylor's series approximation to the function F around the current estimate x_k:

$$F(x_k + \delta) \approx G(\delta) := F(x_k) + \nabla F(x_k)\delta. \tag{D.3}$$

Above, $\nabla F(x)$ denotes the *Jacobian matrix* of the function F, i.e., $\nabla F(x)$ has rows $(\nabla f_1(x))^T, \ldots, (\nabla f_n(x))^T$, the transposed gradients of the functions f_1 through f_n. We denote the components of the n-dimensional vector x using superscripts, i.e. $x =$

$(x^1, \ldots, x^n)$. Let us make these statements more precise:

$$\nabla F(x^1, \ldots, x^n) \;=\; \begin{bmatrix} \frac{\partial f_1}{\partial x^1} & \cdots & \frac{\partial f_1}{\partial x^n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial x^1} & \cdots & \frac{\partial f_n}{\partial x^n} \end{bmatrix}.$$

As before, $G(\delta)$ is the linear approximation to the function F by the hyperplane that is tangent to it at the current point x_k. The next step is to find the value of δ that would make the approximation equal to zero, i.e., the value that satisfies:

$$F(x_k) + \nabla F(x_k)\delta \;=\; \mathbf{0}.$$

Notice that the $\mathbf{0}$ on the RHS is a vector of zeros. If $\nabla F(x_k)$ is nonsingular, the equality above has a (unique) solution given by

$$\delta \;=\; -\nabla F(x_k)^{-1} F(x_k),$$

and the formula for the Newton update in this case is:

$$x_{k+1} \;=\; x_k + \delta = x_k - \nabla F(x_k)^{-1} F(x_k).$$

Example D.2 *Consider the following problem:*

$$F(x) = F(x^1, x^2) = \begin{pmatrix} f_1(x^1, x^2) \\ f_2(x^1, x^2) \end{pmatrix} \;=\; \begin{pmatrix} x^1 x^2 - 2x^1 + x^2 - 2 \\ (x^1)^2 + 2x^1 + (x^2)^2 - 7x^2 + 7 \end{pmatrix} = \mathbf{0}$$

First we calculate the Jacobian:

$$\nabla F(x^1, x^2) \;=\; \begin{pmatrix} x^2 - 2 & x^1 + 1 \\ 2x^1 + 2 & 2x^2 - 7 \end{pmatrix}.$$

If our initial estimate of the solution is $x_0 = (0, 0)$, then the next point generated by Newton's method will be:

$$\begin{aligned} (x_1^1, x_1^2) \;&=\; (x_0^1, x_0^2) - \begin{pmatrix} x_0^2 - 2 & x_0^1 + 1 \\ 2x_0^1 + 2 & 2x_0^2 - 7 \end{pmatrix}^{-1} \begin{pmatrix} x_0^1 x_0^2 - 2x_0^1 + x_0^2 - 2 \\ (x_0^1)^2 + 2x_0^1 + (x_0^2)^2 - 7x_0^2 + 7 \end{pmatrix} \\ &=\; (0, 0) - \begin{pmatrix} -2 & 1 \\ 2 & -7 \end{pmatrix}^{-1} \begin{pmatrix} -2 \\ 7 \end{pmatrix} \\ &=\; (0, 0) - (\frac{7}{12}, -\frac{5}{6}) = (-\frac{7}{12}, \frac{5}{6}). \end{aligned}$$

Appendix E

Karush-Kuhn-Tucker Conditions

Consider an optimization problem given by a nonlinear objective function and/or nonlinear constraints. We can represent such problems in the following generic form:

$$(\mathcal{OP}) \qquad \begin{array}{lrcl} \min_x & f(x) & & \\ & g_i(x) & = & 0, i \in \mathcal{E} \\ & g_i(x) & \geq & 0, i \in \mathcal{I}. \end{array} \tag{E.1}$$

In the remainder of this section we assume that f and $g_i, i \in \mathcal{E} \cup \mathcal{I}$ are all continuously differentiable functions.

One of the most important theoretical issues related to this problem is the identification of necessary and sufficient conditions for optimality. Collectively, these conditions are called the *optimality conditions* and are the subject of this section.

Before presenting the optimality conditions for (E.1) we first discuss a technical condition called *regularity* that is encountered in the theorems that follow:

Definition E.1 *Let x be a vector satisfying $g_i(x) = 0, i \in \mathcal{E}$ and $g_i(x) \geq 0, i \in \mathcal{I}$. Let $\mathcal{J} \subset \mathcal{I}$ be the set of indices for which $g_i(x) \geq 0$ is satisfied with equality. Then, x is a* regular point *of the constraints of (E.1) if the gradient vectors $\nabla g_i(x)$ for $i \in \mathcal{E} \cup \mathcal{J}$ are linearly independent.*

Constraints corresponding to the set $\mathcal{E} \cup \mathcal{J}$ in the definition above, namely, the constraints for which we have $g_i(x) = 0$, are called the *active* constraints at x.

Theorem E.1 (First Order Necessary Conditions) *Let x^* be a local minimizer of the problem (E.1) and assume that x^* is a regular point for the constraints of this problem. Then, there exists $l_i, i \in \mathcal{E} \cup \mathcal{I}$ such that*

$$\nabla f(x^*) - \sum_{i \in \mathcal{E} \cup \mathcal{I}} l_i \nabla g_i(x^*) = 0 \tag{E.2}$$

$$l_i \geq 0, i \in \mathcal{I} \tag{E.3}$$

$$l_i g_i(x^*) = 0, i \in \mathcal{I}. \tag{E.4}$$

First order conditions are satisfied at local minimizers as well as local maximizers and saddle points. When the objective and constraint functions are twice continuously differentiable, one can eliminate maximizers and saddle points using curvature information on the functions.

Theorem E.2 (Second Order Necessary Conditions) *Assume that f and $g_i, i \in \mathcal{E} \cup \mathcal{I}$ are all twice continuously differentiable functions. Let x^* be a local minimizer of the problem (E.1) and assume that x^* is a regular point for the constraints of this problem. Then, there exists $l_i, i \in \mathcal{E} \cup \mathcal{I}$ satisfying (E.2)–(E.4) as well as the following condition:*

$$\nabla^2 f(x^*) - \sum_{i \in \mathcal{E} \cup \mathcal{I}} l_i \nabla^2 g_i(x^*) \tag{E.5}$$

is positive semidefinite on the tangent subspace of active constraints at x^*.

The last part of the theorem above can be restated in terms of the Jacobian of the active constraints. Let $A(x^*)$ denote the Jacobian of the active constraints at x^* and let $N(x^*)$ be a null-space basis for $A(x^*)$. Then, the last condition of the theorem above is equivalent to the following condition:

$$N^T(x^*) \left(\nabla^2 f(x^*) - \sum_{i \in \mathcal{E} \cup \mathcal{I}} l_i \nabla^2 g_i(x^*) \right) N(x^*) \tag{E.6}$$

is positive semidefinite.

The satisfaction of the second order necessary conditions does not always guarantee the local optimality of a given solution vector. The conditions that are sufficient for local optimality are slightly more stringent and a bit more complicated since they need to consider the possibility of *degeneracy*.

Theorem E.3 (Second Order Sufficient Conditions) *Assume that f and $g_i, i \in \mathcal{E} \cup \mathcal{I}$ are all twice continuously differentiable functions. Let x^* be a feasible and regular point for the constraints of the problem (E.1). Let $A(x^*)$ denote the Jacobian of the active constraints at x^* and let $N(x^*)$ be a null-space basis for $A(x^*)$. If there exists $l_i, i \in \mathcal{E} \cup \mathcal{I}$ satisfying (E.2)–(E.4) as well as*

$$g_i(x^*) = 0, i \in \mathcal{I} \text{ implies } l_i > 0, \tag{E.7}$$

and

$$N^T(x^*) \left(\nabla^2 f(x^*) - \sum_{i \in \mathcal{E} \cup \mathcal{I}} l_i \nabla^2 g_i(x^*) \right) N(x^*) \text{ is positive definite} \tag{E.8}$$

then x^* is a local minimizer of the problem (E.1).

The conditions listed in Theorems 1, 2, and 3 are often called Karush-Kuhn-Tucker (KKT) conditions, after their inventors.

Bibliography

[1] V. S. Bawa, S. J. Brown, and R. W. Klein. *Estimation Risk and Optimal Portfolio Choice*. North-Holland, Amsterdam, Netherlands, 1979.

[2] A. Ben-Tal, T. Margalit, and A. N. Nemirovski. Robust modeling of multi-stage portfolio problems. In H. Frenk, K. Roos, T. Terlaky, and S. Zhang, editors, *High Performance Optimization*, pages 303–328. Kluwer Academic Publishers, 2002.

[3] D. R. Cariño, T. Kent, D. H. Myers, C. Stacy, M. Sylvanus, A.L. Turner, K. Watanabe, and W. Ziemba. The Russell-Yasuda Kasai model: An asset/liability model for a Japanese insurance company using multistage stochastic programming. *Interfaces*, 24:29–49, 1994.

[4] M. A. H. Dempster and A. M. Ireland. A financial expert decision support system. In G. Mitra, editor, *Mathematical Models for Decision Support*, volume F48 of *NATO ASI Series*, pages 415–440. 1988.

[5] D. Goldfarb and G. Iyengar. Robust portfolio selection problems. Technical report, IEOR Department, Columbia University, 2001.

[6] B. Halldorsson and R. H. Tütüncü. An interior-point method for a class of saddle point problems. *Journal of Optimization Theory and Applications*, 116(3):559–590, 2003.

[7] S. Herzel. Arbitrage opportunities on derivatives: A linear programming approach. Technical report, Department of Economics, University of Perugia, 2000.

[8] R. O. Michaud. The Markowitz optimization enigma: Is optimized optimal? *Financial Analysts Journal*, 45:31–42, 1989.

[9] R. O. Michaud. *Efficient Asset Management*. Harvard Business School Press, Boston, Massachusetts, 1998.

[10] Yu. Nesterov and A. Nemirovski. *Interior-Point Polynomial Algorithms in Convex Programming*. SIAM, Philadelphia, Pennsylvania, 1994.

[11] M. Pınar. Minimum risk arbitrage with risky financial contracts. Technical report, Bilkent University, Ankara, Turkey, 2001.

[12] R. T. Rockafellar and S. Uryasev. Optimization of conditional value-at-risk. *The Journal of Risk*, 2:21–41, 2000.

[13] W. F. Sharpe. Determining a fund's effective asset mix. *Investment Management Review*, pages 59–69, December 1988.

[14] W. F. Sharpe. Asset allocation: Management style and performance measurement. *Journal of Portfolio Management*, pages 7–19, Winter 1992.

[15] W.F. Sharpe. The Sharpe ratio. *Journal of Portfolio Management*, Fall:49–58, 1994.

[16] R. H. Tütüncü and M. Koenig. Robust asset allocation. Technical report, Department of Mathematical Sciences, Carnegie Mellon University, August 2002. To appear in *Annals of Operations Research*.

[17] S. Uryasev. Conditional value-at-risk: Optimization algorithms and applications. *Financial Engineering News*, 14:1–6, 2000.