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 efficient algorithms. Mathematically, it refers to the minimization (or maximization) of a given *objective function* of several *decision variables* that satisfy functional *constraints*. A typical optimization model addresses the allocation of scarce resources among possible alternative uses in order to maximize an objective function such as total profit.

Decision variables, the objective function, and constraints are three essential elements of any optimization problem. Problems that lack constraints are called *unconstrained optimization* problems, while others are often referred to as *constrained optimization* problems. Problems with no objective functions are called *feasibility* problems. Some problems may have multiple objective functions. These problems are often addressed by reducing them to a single-objective optimization problem or a sequence of such 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. We continue with a list of problem classes that we will encounter in this book.

1.1 Optimization problems

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

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

is called an optimization problem. 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 $x^k \in S$ such that $f(x^k) \to -\infty$ as $k \to +\infty$, then the problem is *unbounded*. If the problem is neither infeasible nor unbounded, then it is often possible to find a solution $x^* \in S$ that satisfies

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

Such an x^* is called a *global minimizer* of the problem (1.1). 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 $x^* \in S$ 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 (1.1). 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

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

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

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

Many factors affect whether optimization problems can be solved efficiently. For example, the number n of decision variables, and the total number of constraints $|\mathcal{E}| + |\mathcal{I}|$, are generally good predictors of how difficult it will be to solve a given optimization problem. Other factors are related to the properties of the functions f and g_i that define the problem. Problems with a linear objective function and linear constraints are easier, as are problems with convex objective functions and convex feasible sets. For this reason, instead of general purpose optimization algorithms, researchers have developed different algorithms for problems with special characteristics. We list the main types of optimization problems we will encounter. A more complete list can be found, for example, on the *Optimization Tree* available from www-fp.mcs.anl.gov/otc/Guide/OptWeb/.

1.1.1 Linear and nonlinear programming

One of the most common and easiest optimization problems is *linear optimization* or *linear programming* (LP). This is the problem of optimizing a linear objective function subject to linear equality and inequality constraints. It corresponds to the case where the functions f and g_i in (1.2) are all linear. If either f or one of the functions g_i is not linear, then the resulting problem is a *nonlinear programming* (NLP) problem.

The standard form of the LP is given below:

$$\begin{aligned} \min_x \ & c^{\mathrm{T}}x \\ & Ax = b \\ & x \geq 0, \end{aligned} \tag{1.3}$$

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. In this book, a k-vector is also viewed as a $k \times 1$ matrix. For an $m \times n$ matrix M, the notation M^{T} denotes the *transpose* matrix, namely the $n \times m$ matrix with entries $M^{\mathrm{T}}_{ij} = M_{ji}$. As an example, in the above formulation c^{T} is a $1 \times n$ matrix and $c^{\mathrm{T}}x$ is the 1×1 matrix with entry $\sum_{j=1}^n c_j x_j$. The objective in (1.3) is to minimize the linear function $\sum_{j=1}^n c_j x_j$.

As with (1.1), the problem (1.3) is said to be *feasible* if its constraints are consistent (i.e., they define a nonempty region) and it is called *unbounded* if there exists a sequence of feasible vectors $\{x^k\}$ such that $c^{\mathrm{T}}x^k \to -\infty$. When (1.3) 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. Similar definitions apply to nonlinear programming problems.

The best known and most successful methods for solving LPs are the simplex and interior-point methods. NLPs can be solved using gradient search techniques as well as approaches based on Newton's method such as interior-point and sequential quadratic programming methods.

1.1.2 Quadratic programming

A more general optimization problem is the *quadratic optimization* 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:

$$\begin{aligned} \min_x \ & \tfrac{1}{2}x^{\mathrm{T}}Qx + c^{\mathrm{T}}x \\ & Ax = b \\ & x \geq 0, \end{aligned} \tag{1.4}$$

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 $x^{\mathrm{T}}Qx = \frac{1}{2}x^{\mathrm{T}}(Q + Q^{\mathrm{T}})x$, one can assume without loss of generality that Q is *symmetric*, i.e., $Q_{ij} = Q_{ji}$.

The objective function of the problem (1.4) is a convex function of x when Q is a *positive semidefinite matrix*, i.e., when $y^{\mathrm{T}}Qy \geq 0$ for all y (see Appendix A for a discussion on convex functions). This condition is equivalent to Q having only 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. Here we are referring to a classical notion used to measure computational complexity. Polynomial time algorithms are efficient in the sense that they always find an optimal solution in an amount of time that is guaranteed to be at most a polynomial function of the input size.

1.1.3 Conic optimization

Another generalization of (1.3) is obtained when the nonnegativity constraints $x \geq 0$ are replaced by general conic inclusion constraints. This is called a *conic optimization* (CO) problem. For this purpose, we consider a closed convex cone C (see Appendix B for a brief discussion on cones) in a finite-dimensional vector space X and the following conic optimization problem:

$$\begin{aligned} \min_x \; & c^{\mathrm{T}}x \\ & Ax = b \\ & x \in C. \end{aligned} \tag{1.5}$$

When $X = I\!R^n$ and $C = I\!R^n_+$, 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:

$$C_q := \left\{x = (x_1, x_2, \ldots, x_n) \in I\!R^n : x_1^2 \geq x_2^2 + \cdots + x_n^2, x_1 \geq 0\right\},$$

and the cone of symmetric positive semidefinite matrices:

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

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

1.1.4 Integer programming

Integer programs are optimization problems that require some or all of the variables to take integer values. This restriction on the variables often makes the problems very hard to solve. Therefore we will focus on *integer linear programs*, which have a linear objective function and linear constraints. A *pure* integer linear program (ILP) is given by:

$$\begin{aligned} \min_x \; & c^{\mathrm{T}} x \\ & Ax \geq b \\ & x \geq 0 \text{ and integral,} \end{aligned} \tag{1.6}$$

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

An important case occurs when the variables x_j represent binary decision variables, that is, $x \in \{0, 1\}^n$. The problem is then called a *0–1 linear program*.

When there are both continuous variables and integer constrained variables, the problem is called a *mixed integer linear program* (MILP):

$$\begin{aligned} \min_x \; & c^{\mathrm{T}} x \\ & Ax \geq b \\ & x \geq 0 \\ & x_j \in I\!N \text{ for } j = 1, \ldots, p. \end{aligned} \tag{1.7}$$

where A, b, c are given data and the integer p (with $1 \leq p < n$) is also part of the input.

1.1.5 Dynamic programming

Dynamic programming refers to a computational method involving recurrence relations. This technique was developed by Richard Bellman in the early 1950s. It arose from studying programming problems in which changes over time were important, thus the name "dynamic programming." However, the technique can also be applied when time is not a relevant factor in the problem. The idea is to divide the problem into "stages" in order to perform the optimization recursively. It is possible to incorporate stochastic elements into the recursion.

1.2 Optimization with data uncertainty

In all the problem classes discussed so far (except dynamic programming), we made the implicit assumption that the *data* of the problem, namely the parameters 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 cannot be known exactly at the time the problem must be formulated and solved. Such situations are especially common in models involving 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 one wants a solution that behaves well in all possible realizations of the uncertain data. 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 programming

The term *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:

$$\begin{array}{lll} \max_x & a^{\mathrm{T}}x + E[\max_{y(\omega)} c(\omega)^{\mathrm{T}}y(\omega)] & \\ & Ax & = b \\ & B(\omega)x + & C(\omega)y(\omega) = d(\omega) \\ & x \geq 0, & y(\omega) \geq 0, \end{array} \tag{1.8}$$

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

Note that, once the first-stage decisions x 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:

$$\begin{array}{lll} f(x,\omega) = \max & c(\omega)^{\mathrm{T}}y(\omega) & \\ & C(\omega)y(\omega) = d(\omega) - B(\omega)x \\ & y(\omega) \geq 0. \end{array} \tag{1.9}$$

Let $f(x) = E[f(x, \omega)]$ denote the expected value of the optimal value of this problem. Then, the two-stage stochastic linear program becomes

$$\begin{aligned} \max\ & a^{\mathrm{T}}x + f(x) \\ & Ax = b \\ & x \geq 0. \end{aligned} \tag{1.10}$$

Thus, if the (possibly nonlinear) function $f(x)$ is known, the problem reduces to a nonlinear programming problem. When the data $c(\omega)$, $B(\omega)$, $C(\omega)$, and $d(\omega)$ 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 can be realized by the uncertain parameters.

There are different definitions and interpretations of robustness and the resulting models differ accordingly. One important concept is *constraint robustness*, often called *model robustness* in the literature. This refers to solutions that remain *feasible* for all possible values of the uncertain inputs. This type of solution is required in several engineering applications. Here is an example adapted from Ben-Tal and Nemirovski [8]. 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 cannot exceed what is used in that phase plus what is left over for the next phase). The quantities of the end products of a particular phase may depend on external, uncontrollable factors and are therefore uncertain. However, no matter what the values of these uncontrollable factors are, the balance constraints *must* be satisfied. Therefore, the solution must be constraint robust with respect to the uncertainties of the problem. A mathematical model for finding constraint-robust solutions will be described. First, consider an optimization problem of the form:

$$\begin{aligned} \min_x\ & f(x) \\ & G(x, p) \in K. \end{aligned} \tag{1.11}$$

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 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:

$$\begin{array}{ll} \min_x & f(x) \\ & G(x, p) \in K, \quad \forall p \in \mathcal{U}. \end{array} \tag{1.12}$$

A related concept is *objective robustness*, which occurs when uncertain parameters appear in the objective function. This is often referred to as solution robustness in the literature. Such robust solutions must remain close to optimal for all possible realizations of the uncertain parameters. Next, consider an optimization problem of the form:

$$\begin{array}{ll} \min_x & f(x, p) \\ & x \in S. \end{array} \tag{1.13}$$

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 is obtained by solving:

$$\min_{x \in S} \max_{p \in \mathcal{U}} f(x, p). \tag{1.14}$$

Note that objective robustness is a special case of constraint robustness. Indeed, by introducing a new variable t (to be minimized) into (1.13) and imposing the constraint $f(x, p) \leq t$, we get an equivalent problem to (1.13). The constraint-robust formulation of the resulting problem is equivalent to (1.14).

Constraint robustness and objective 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 [54] in the 1950s and the option pricing formulas developed by Black, Scholes, and Merton [15, 55] in the late 1960s and early 1970s. 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 introduce 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 1950s. His work formalized the diversification principle in portfolio selection and, as mentioned above, earned him the 1990 Nobel prize for Economics. 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 $i = 1, \ldots, n$, 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:

$$E[x] = x_1\mu_1 + \cdots + x_n\mu_n = \mu^{\mathrm{T}}x,$$

and

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

where $\rho_{ii} \equiv 1$, $Q_{ij} = \rho_{ij}\sigma_i\sigma_j$, 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 optimization 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 R of expected return. Mathematically, this formulation produces a convex quadratic programming problem:

$$\begin{aligned} \min_x \; & x^{\mathrm{T}}Qx \\ & e^{\mathrm{T}}x = 1 \\ & \mu^{\mathrm{T}}x \geq R \\ & x \geq 0, \end{aligned} \tag{1.15}$$

where e is an n-dimensional vector with all components equal to 1. 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 rule out short sales (selling a security that you do not have). Note that the matrix Q is positive semidefinite since $x^{\mathrm{T}}Qx$, the variance of the portfolio, must be nonnegative for every portfolio (feasible or not) x.

As an alternative to problem (1.15), we may choose to maximize the expected return of a portfolio while limiting the variance of its return. Or, we can maximize a risk-adjusted expected return, which is defined as the expected return minus a multiple of the variance. These two formulations are essentially equivalent to (1.15), as we will see in Chapter 8.

The model (1.15) 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 investor preferences 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 significantly harder to solve.

Asset allocation problems have the same mathematical structure as 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 asset classes are large capitalization stocks, small capitalization stocks, foreign stocks, government bonds, corporate bonds, etc. There are many mutual funds focusing on specific asset classes and one can therefore conveniently invest in these asset classes by purchasing the relevant mutual funds. After estimating the expected returns, variances, and covariances for different asset classes, one can formulate a QP identical to (1.15) and obtain efficient portfolios of these asset classes.

A different strategy for portfolio selection is to try to mirror the movements of a broad market population using a significantly smaller number of securities. Such a portfolio is called an index fund. No effort is made to identify mispriced securities. The assumption is that the market is efficient and therefore no superior risk-adjusted returns can be achieved by stock picking strategies since the stock prices reflect all the information available in the marketplace. Whereas actively managed funds incur transaction costs that reduce their overall performance, index funds are not actively traded and incur low management fees. They are typical of a passive management strategy. How do investment companies construct index funds? There are numerous ways of doing this. One way is to solve a clustering problem