最佳化方法與應用
MA5037-*

Chapter 1. Introduction

Introduction

Optimization is the minimization or maximization of a function subject to constraints on its variables. We use the following notation:

1. x is the vector of variables, also called unknowns or parameters;
2. f is the objective function, a (scalar) function of x that we want to maximize or minimize;
3. c_i are constraint functions, which are scalar functions of x that define certain equations and inequalities that the unknown vector x must satisfy.

Using this notation, the optimization problem can be written as

$$\min_{x\in\mathbb{R}^n} f(x) \quad \text{subject to} \quad \begin{cases} c_i(x) = 0 & \text{if } i \in \mathcal{E}\,, \\ c_i(x) \geqslant 0 & \text{if } i \in \mathcal{I}\,. \end{cases} \tag{1}$$

Here $\mathcal{E}$ and $\mathcal{I}$ are sets of indices for equality and inequality constraints, respectively.

Introduction

Optimization is the minimization or maximization of a function subject to constraints on its variables. We use the following notation:

1. x is the vector of variables, also called unknowns or parameters;
2. f is the objective function, a (scalar) function of x that we want to maximize or minimize;
3. c_i are constraint functions, which are scalar functions of x that define certain equations and inequalities that the unknown vector x must satisfy.

Using this notation, the optimization problem can be written as

$$\min_{x \in \mathbb{R}^n} f(x) \quad \text{subject to} \quad \begin{cases} c_i(x) = 0 & \text{if } i \in \mathcal{E}\,, \\ c_i(x) \geqslant 0 & \text{if } i \in \mathcal{I}\,. \end{cases} \tag{1}$$

Here $\mathcal{E}$ and $\mathcal{I}$ are sets of indices for equality and inequality constraints, respectively.

Introduction

Often it is more natural or convenient to label the unknowns with two or three subscripts, or to refer to different variables by completely different names, so that relabeling is necessary to pose the problem in the form (1). Another common difference is that we are required to maximize rather than minimize f, but we can accommodate this change easily by minimizing $-f$ in the formulation (1). Good modeling systems perform the conversion to standardized formulations such as (1) transparently to the user.

Introduction

- **Continuous v.s. Discrete Optimization**:

In some optimization problems the variables make sense only if they take on integer values. The mathematical formulation of such problems includes integrality constraints, which have the form $x_i \in \mathbb{Z}$, where $\mathbb{Z}$ is the set of integers, or binary constraints, which have the form $x_i \in \{0, 1\}$, in addition to algebraic constraints like those appearing in (1). Problems of this type are called **integer programming problems**. If some of the variables in the problem are not restricted to be integer or binary variables, they are sometimes called **mixed integer programming problems**, or **MIP**s for short.

Introduction

- **Continuous v.s. Discrete Optimization**:

In some optimization problems the variables make sense only if they take on integer values. The mathematical formulation of such problems includes integrality constraints, which have the form $x_i \in \mathbb{Z}$, where $\mathbb{Z}$ is the set of integers, or binary constraints, which have the form $x_i \in \{0, 1\}$, in addition to algebraic constraints like those appearing in (1). Problems of this type are called **integer programming problems**. If some of the variables in the problem are not restricted to be integer or binary variables, they are sometimes called **mixed integer programming problems**, or **MIP**s for short.

Introduction

Integer programming problems are a type of discrete optimization problem. Generally, discrete optimization problems may contain not only integers and binary variables, but also more abstract variable objects such as permutations of an ordered set. The defining feature of a discrete optimization problem is that the unknown x is drawn from a finite (but often very large) set.

Introduction

Integer programming problems are a type of discrete optimization problem. Generally, discrete optimization problems may contain not only integers and binary variables, but also more abstract variable objects such as permutations of an ordered set. The defining feature of a discrete optimization problem is that the unknown x is drawn from a finite (but often very large) set.

Introduction

By contrast, the feasible set for continuous optimization problems, the class of problems studied in this course, is usually uncountably infinite, as when the components of x are allowed to be real numbers. Continuous optimization problems are easier to solve because the smoothness of the functions makes it possible to use objective and constraint information at a particular point x to deduce information about the function's behavior at all points close to x.

In discrete problems, by constrast, the behavior of the objective and constraints may change significantly as we move from one feasible point to another, even if the two points are "close" by some measure. The feasible sets for discrete optimization problems can be thought of as exhibiting an extreme form of non-convexity, as a convex combination of two feasible points is in general not feasible.

Introduction

By contrast, the feasible set for continuous optimization problems, the class of problems studied in this course, is usually uncountably infinite, as when the components of x are allowed to be real numbers. Continuous optimization problems are easier to solve because the smoothness of the functions makes it possible to use objective and constraint information at a particular point x to deduce information about the function's behavior at all points close to x.

In discrete problems, by constrast, the behavior of the objective and constraints may change significantly as we move from one feasible point to another, even if the two points are "close" by some measure. The feasible sets for discrete optimization problems can be thought of as exhibiting an extreme form of non-convexity, as a convex combination of two feasible points is in general not feasible.

Introduction

By contrast, the feasible set for continuous optimization problems, the class of problems studied in this course, is usually uncountably infinite, as when the components of x are allowed to be real numbers. Continuous optimization problems are easier to solve because the smoothness of the functions makes it possible to use objective and constraint information at a particular point x to deduce information about the function's behavior at all points close to x.

In discrete problems, by constrast, the behavior of the objective and constraints may change significantly as we move from one feasible point to another, even if the two points are "close" by some measure. The feasible sets for discrete optimization problems can be thought of as exhibiting an extreme form of non-convexity, as a convex combination of two feasible points is in general not feasible.

Introduction

By contrast, the feasible set for continuous optimization problems, the class of problems studied in this course, is usually uncountably infinite, as when the components of x are allowed to be real numbers. Continuous optimization problems are easier to solve because the smoothness of the functions makes it possible to use objective and constraint information at a particular point x to deduce information about the function's behavior at all points close to x.

In discrete problems, by constrast, the behavior of the objective and constraints may change significantly as we move from one feasible point to another, even if the two points are "close" by some measure. The feasible sets for discrete optimization problems can be thought of as exhibiting an extreme form of non-convexity, as a convex combination of two feasible points is in general not feasible.

Introduction

- **Constrained and Unconstrained Optimizations**:

Problems with the general form (1) can be classified according to the nature of the objective function and constraints (linear, nonlinear, convex), the number of variables (large or small), the smoothness of the functions (differentiable or non-differentiable), and so on. An important distinction is between problems that have constraints on the variables and those that do not. This textbook is divided into two parts according to this classification.

Introduction

Unconstrained optimization problems, for which we have $\mathcal{E} = \mathcal{I} = \varnothing$ in (1), arise directly in many practical applications. Even for some problems with natural constraints on the variables, it may be safe to disregard them as they do not affect on the solution and do not interfere with algorithms. Unconstrained problems arise also as reformulations of constrained optimization problems, in which the constraints are replaced by penalization terms added to objective function that have the effect of discouraging constraint violations.

Introduction

Unconstrained optimization problems, for which we have $\mathcal{E} = \mathcal{I} = \varnothing$ in (1), arise directly in many practical applications. Even for some problems with natural constraints on the variables, it may be safe to disregard them as they do not affect on the solution and do not interfere with algorithms. Unconstrained problems arise also as reformulations of constrained optimization problems, in which the constraints are replaced by penalization terms added to objective function that have the effect of discouraging constraint violations.

Introduction

Unconstrained optimization problems, for which we have $\mathcal{E} = \mathcal{I} = \emptyset$ in (1), arise directly in many practical applications. Even for some problems with natural constraints on the variables, it may be safe to disregard them as they do not affect on the solution and do not interfere with algorithms. Unconstrained problems arise also as reformulations of constrained optimization problems, in which the constraints are replaced by penalization terms added to objective function that have the effect of discouraging constraint violations.

Introduction

Constrained optimization problems arise from models in which constraints play an essential role, for example in imposing budgetary constraints in an economic problem or shape constraints in a design problem. These constraints may be simple bounds such as $0 \leqslant x_1 \leqslant 100$, more general linear constraints such as $\sum_i x_i \leqslant 1$, or nonlinear inequalities that represent complex relationships among the variables.

Introduction

When the objective function and all the constraints are linear functions of x, the problem is a **linear programming problem**. Problems of this type are probably the most widely formulated and solved of all optimization problems, particularly in management, financial, and economic applications. Nonlinear programming problems, in which at least some of the constraints or the objective are nonlinear functions, tend to arise naturally in the physical sciences and engineering, and are becoming more widely used in management and economic sciences as well.

Introduction

When the objective function and all the constraints are linear functions of x, the problem is a **linear programming problem**. Problems of this type are probably the most widely formulated and solved of all optimization problems, particularly in management, financial, and economic applications. Nonlinear programming problems, in which at least some of the constraints or the objective are nonlinear functions, tend to arise naturally in the physical sciences and engineering, and are becoming more widely used in management and economic sciences as well.

Introduction

- **Global and Local Optimization**:

Many algorithms for nonlinear optimization problems seek only a local solution, a point at which the objective function is smaller than at all other feasible nearby points. They do not always find the global solution, which is the point with lowest function value among all feasible points. Global solutions are needed in some applications, but for many problems they are difficult to recognize and even more difficult to locate. For convex programming problems, and more particularly for linear programs, local solutions are also global solutions. General nonlinear problems, both constrained and unconstrained, may possess local solutions that are not global solutions.

Introduction

- **Global and Local Optimization**:

Many algorithms for nonlinear optimization problems seek only a local solution, a point at which the objective function is smaller than at all other feasible nearby points. They do not always find the global solution, which is the point with lowest function value among all feasible points. Global solutions are needed in some applications, but for many problems they are difficult to recognize and even more difficult to locate. For convex programming problems, and more particularly for linear programs, local solutions are also global solutions. General nonlinear problems, both constrained and unconstrained, may possess local solutions that are not global solutions.

Introduction

In this textbook we treat global optimization only in passing and focus instead on the computation and characterization of local solutions. We note, however, that many successful global optimization algorithms require the solution of many local optimization problems, to which the algorithms described in this textbook can be applied.

Introduction

- **Stochastic and Deterministic Optimization**:

In some optimization problems, the model cannot be fully specified because it depends on quantities that are unknown at the time of formulation. This characteristic is shared by many economic and financial planning models, which may depend for example on future interest rates, future demands for a product, or future commodity prices, but uncertainty can arise naturally in almost any type of application.

Introduction

Rather than just use a “best guess” for the uncertain quantities, modelers may obtain more useful solutions by incorporating additional knowledge about these quantities into the model. For example, they may know a number of possible scenarios for the uncertain demand, along with estimates of the probabilities of each scenario. Stochastic optimization algorithms use these quantifications of the uncertainty to produce solutions that optimize the expected performance of the model.

Introduction

Related paradigms for dealing with uncertain data in the model include chance constrained optimization, in which we ensure that the variables x satisfy the given constraints to some specified probability, and robust optimization, in which certain constraints are required to hold for all possible values of the uncertain data.

We do not consider stochastic optimization problems further in this textbook, focusing instead on deterministic optimization problems, in which the model is completely known. Many algorithms for stochastic optimization do, however, proceed by formulating one or more deterministic subproblems, each of which can be solved by the techniques outlined here.

Introduction

- **Convexity**:

The concept of convexity is fundamental in optimization. Many practical problems possess this property, which generally makes them easier to solve both in theory and practice.

The term "convex" can be applied both to sets and to functions. A set $S \in \mathbb{R}^n$ is a convex set if the straight line segment connecting any two points in S lies entirely inside S. Formally, for any two points $x, y \in S$, we have $\alpha x + (1-\alpha)y \in S$ for all $\alpha \in [0,1]$. A function f is a convex function if **its domain S is a convex set** and if for any two points x and y in S, the following property is satisfied:

$$f(\alpha x + (1-\alpha)y) \leqslant \alpha f(x) + (1-\alpha)f(y) \quad \forall\, \alpha \in [0,1].$$

Introduction

- **Convexity**:

The concept of convexity is fundamental in optimization. Many practical problems possess this property, which generally makes them easier to solve both in theory and practice.

The term "convex" can be applied both to sets and to functions. A set $S \in \mathbb{R}^n$ is a convex set if the straight line segment connecting any two points in S lies entirely inside S. Formally, for any two points $x, y \in S$, we have $\alpha x + (1-\alpha)y \in S$ for all $\alpha \in [0,1]$. A function f is a convex function if **its domain S is a convex set** and if for any two points x and y in S, the following property is satisfied:

$$f(\alpha x + (1-\alpha)y) \leqslant \alpha f(x) + (1-\alpha)f(y) \quad \forall\, \alpha \in [0,1].$$

Introduction

Example

1. Any ball in $\mathbb{R}^n$ is convex.
2. Any polyhedron, which is a set defined by linear equalities and inequalities; that is,

$$\left\{x \in \mathbb{R}^n \,\middle|\, Ax = b, Cx \leqslant d\right\},$$

where A and C are matrices of appropriate dimension, and b and d are vectors, is convex.

Example

1. For any constant vector $c \in \mathbb{R}^n$ and scalar α, the linear function $f(x) = c^{\mathrm{T}}x + \alpha$ is convex.
2. For any symmetric positive semi-definite matrix H, the quadratic function $f(x) = x^{\mathrm{T}}Hx$ is convex.

Introduction

We say that f is strictly convex if

$$f(\alpha x+(1-\alpha)y) < \alpha f(x)+(1-\alpha)f(y) \quad \forall\, \alpha \in (0,1).$$

A function f is said to be concave if $-f$ is convex. If the objective function in the optimization problem (1) and the feasible region are both convex, then any local solution of the problem is in fact a global solution. The term **convex programming** is used to describe a special case of the general constrained optimization problem (1) in which

1. the objective function is convex,
2. the equality constraint functions $c_i(\cdot)$, $i \in \mathcal{E}$, are linear, and
3. the inequality constraint functions $c_i(\cdot)$, $i \in I$, are concave.

Introduction

We say that f is strictly convex if

$$f(\alpha x + (1-\alpha)y) < \alpha f(x) + (1-\alpha)f(y) \quad \forall\, \alpha \in (0,1).$$

A function f is said to be concave if $-f$ is convex. If the objective function in the optimization problem (1) and the feasible region are both convex, then any local solution of the problem is in fact a global solution. The term **convex programming** is used to describe a special case of the general constrained optimization problem (1) in which

1. the objective function is convex,
2. the equality constraint functions $c_i(\cdot)$, $i \in \mathcal{E}$, are linear, and
3. the inequality constraint functions $c_i(\cdot)$, $i \in I$, are concave.

Introduction

We say that f is strictly convex if

$$f(\alpha x + (1-\alpha)y) < \alpha f(x) + (1-\alpha)f(y) \quad \forall\, \alpha \in (0,1).$$

A function f is said to be concave if $-f$ is convex. If the objective function in the optimization problem (1) and the feasible region are both convex, then any local solution of the problem is in fact a global solution. The term **convex programming** is used to describe a special case of the general constrained optimization problem (1) in which

1. the objective function is convex,
2. the equality constraint functions $c_i(\cdot)$, $i \in \mathcal{E}$, are linear, and
3. the inequality constraint functions $c_i(\cdot)$, $i \in I$, are concave.

Introduction

If we write our optimization problem as

$$\min_{x\in\mathbb{R}^n} f(x) \quad \text{subject to} \quad \begin{cases} c_i(x) = 0 & \text{if } i \in \mathcal{E}\,, \\ c_i(x) \leqslant 0 & \text{if } i \in \mathcal{I}\,. \end{cases} \tag{1'}$$

Then the term **convex programming** is used to describe a special case of the general constrained optimization problem $(1')$ in which

1. the objective function is convex,
2. the equality constraint functions $c_i(\cdot)$, $i \in \mathcal{E}$, are linear, and
3. the inequality constraint functions $c_i(\cdot)$, $i \in I$, are convex.

Introduction

- **Optimization Algorithm**:

Optimization algorithms are iterative. They begin with an initial guess of the variable x and generate a sequence of improved estimates (called "iterates") until they terminate, hopefully at a solution. The strategy used to move from one iterate to the next distinguishes one algorithm from another. Most strategies make use of the values of the objective function f, the constraint functions c_i, and possibly the first and second derivatives of these functions. Some algorithms accumulate information gathered at previous iterations, while others use only local information obtained at the current point.

Introduction

- **Optimization Algorithm**:

Optimization algorithms are iterative. They begin with an initial guess of the variable x and generate a sequence of improved estimates (called "iterates") until they terminate, hopefully at a solution. The strategy used to move from one iterate to the next distinguishes one algorithm from another. Most strategies make use of the values of the objective function f, the constraint functions c_i, and possibly the first and second derivatives of these functions. Some algorithms accumulate information gathered at previous iterations, while others use only local information obtained at the current point.

Introduction

Good algorithms should possess the following properties:

1. **Robustness**: They should perform well on a wide variety of problems in their class, for all reasonable values of the starting point.
2. **Efficiency**: They should not require excessive computer time or storage.
3. **Accuracy**: They should be able to identify a solution with precision, without being overly sensitive to errors in the data or to the arithmetic rounding errors that occur when the algorithm is implemented on a computer.

Introduction

These goals may conflict. For example, a rapidly convergent method for a large unconstrained nonlinear problem may require too much computer storage. On the other hand, a robust method may also be the slowest. Tradeoffs between convergence rate and storage requirements, and between robustness and speed, and so on, are central issues in numerical optimization.

The mathematical theory of optimization is used both to characterize optimal points and to provide the basis for most algorithms. It is not possible to have a good understanding of numerical optimization without a firm grasp of the supporting theory. Accordingly, the course gives a solid (though not comprehensive) treatment of optimality conditions, as well as convergence analysis that reveals the strengths and weaknesses of some of the most important algorithms.

Introduction

These goals may conflict. For example, a rapidly convergent method for a large unconstrained nonlinear problem may require too much computer storage. On the other hand, a robust method may also be the slowest. Tradeoffs between convergence rate and storage requirements, and between robustness and speed, and so on, are central issues in numerical optimization.

The mathematical theory of optimization is used both to characterize optimal points and to provide the basis for most algorithms. It is not possible to have a good understanding of numerical optimization without a firm grasp of the supporting theory. Accordingly, the course gives a solid (though not comprehensive) treatment of optimality conditions, as well as convergence analysis that reveals the strengths and weaknesses of some of the most important algorithms.

Introduction

- **Side Note**:

Optimization traces its roots to the **calculus of variations**（變分學）and the work of **Euler** and **Lagrange**. The development of linear programming in the 1940s broadened the field and stimulated much of the progress in modern optimization theory and practice during the past 60 years.

Optimization is often called **mathematical programming**, a somewhat confusing term coined in the 1940s, before the word "programming" became inextricably linked with computer software. The original meaning of this word was more inclusive（包容性）, with connotations（內涵）of algorithm design and analysis.

Modeling will not be treated extensively in this course.

Introduction

- **Side Note**:

Optimization traces its roots to the **calculus of variations**（變分學）and the work of **Euler** and **Lagrange**. The development of linear programming in the 1940s broadened the field and stimulated much of the progress in modern optimization theory and practice during the past 60 years.

Optimization is often called **mathematical programming**, a somewhat confusing term coined in the 1940s, before the word "programming" became inextricably linked with computer software. The original meaning of this word was more inclusive（包容性）, with connotations（內涵）of algorithm design and analysis.

Modeling will not be treated extensively in this course.

Introduction

- **Side Note**:

Optimization traces its roots to the **calculus of variations**（變分學）and the work of **Euler** and **Lagrange**. The development of linear programming in the 1940s broadened the field and stimulated much of the progress in modern optimization theory and practice during the past 60 years.

Optimization is often called **mathematical programming**, a somewhat confusing term coined in the 1940s, before the word “programming” became inextricably linked with computer software. The original meaning of this word was more inclusive（包容性）, with connotations（內涵）of algorithm design and analysis.

Modeling will not be treated extensively in this course.