FERHAT ABBAS University (Sétif 1)

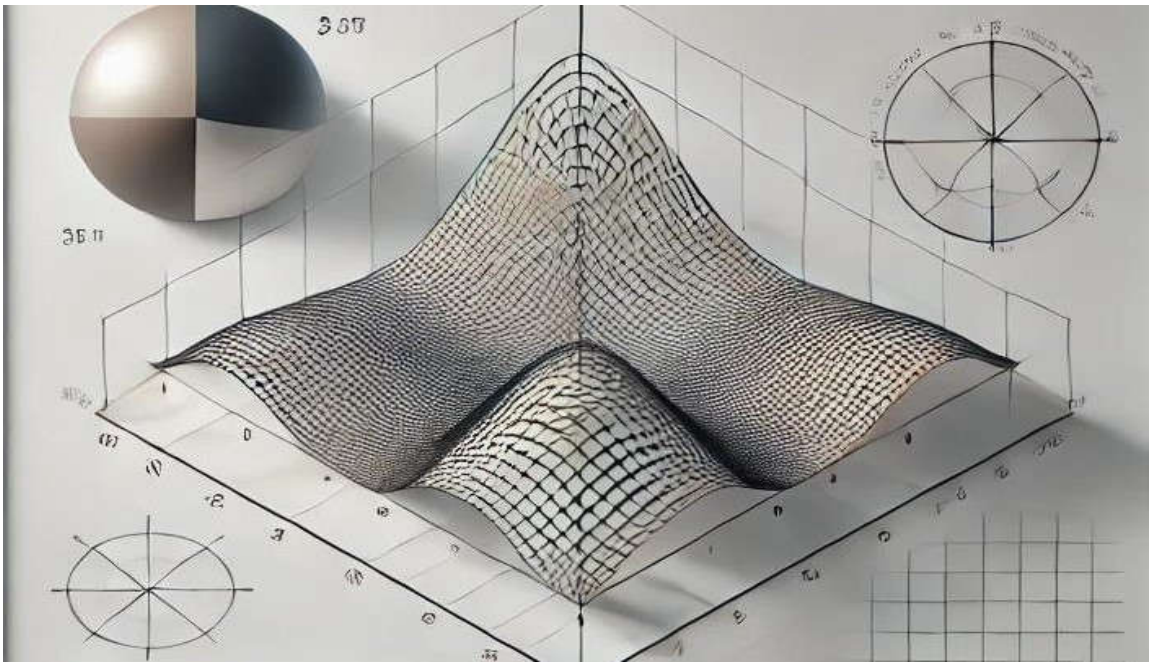
Faculty of Economics, Commerce and Management Sciences

Department of Economics

Optimization Methods

(Non-Linear Programming)

موجهة لطلبة السنة الثانية ماستر تخصص : اقتصاد كمي



Course for Students in Master1, Option: Quantitative Economics

2024/2025

Dr. Fatih CHELLAI

Course Content
Ch1 optimization and mathematical programming (introduction)
Ch2 Convex sets and convex functions
Ch3 multidimensional convex functions
Ch4 identification of optimum of a function
Ch5 Non linear optimization without constraints (newthon method)
Ch6 Steepest Ascent method
Ch7 Jacobean Method
Ch8 Lagrange Method
Ch9 Quadratic Programming
Ch10 Applications using R , Lindo, QM4Windows..

FERHAT ABBAS University (Sétif 1)
Faculty of Economics, Commerce and Management Sciences
Department of Economics

Optimization Methods (Non-Linear Programming)

Course for Students in Master 1, Option: Quantitative Economics

Dr. Fatih CHELLAI

Introduction

Optimization is a fundamental discipline in applied mathematics and economics that focuses on finding the best possible solutions to problems involving constraints and objectives. It plays a crucial role in various fields, including economics, finance, operations research, engineering, and machine learning. This course, "Optimization Methods (Non-Linear Programming)," aims to provide students with the theoretical foundation and practical skills needed to analyze and solve optimization problems, particularly in the context of quantitative economics.

Objective of the Course

The primary objective of this course is to introduce students to the principles and methodologies of optimization, with a focus on nonlinear programming techniques. Students will gain a deep understanding of mathematical programming, convexity, optimization techniques, and their applications in economic and managerial decision-making. By the end of this course, students will be able to:

- Grasp the fundamental concepts of optimization and mathematical programming.
- Understand convex sets, convex functions, and their properties.
- Identify optima of functions using analytical and numerical methods.
- Apply various optimization techniques, including Newton's method, the Steepest Ascent method, the Jacobean method, and the Lagrange method.
- Solve quadratic programming problems and real-world optimization applications.

- Utilize computational tools such as R, Lindo, and QM4Windows for solving optimization problems.

Relevance to Quantitative Economics

Optimization methods are essential for economic analysis, as they provide powerful tools for decision-making in uncertain and constrained environments. Economic models often require optimization techniques to determine equilibrium states, maximize utility, minimize costs, and optimize resource allocation. In quantitative economics, these methods help in:

- Estimating economic parameters using nonlinear regression models.
- Solving constrained optimization problems in microeconomics and macroeconomics.
- Analyzing economic efficiency and production functions.
- Developing predictive models for market behavior and economic forecasting.

Structure of the Course

The course is structured into ten comprehensive chapters covering both theoretical foundations and computational applications:

1. **Optimization and Mathematical Programming (Introduction) –** Overview of optimization problems, classification, and applications.
2. **Convex Sets and Convex Functions –** Definition, properties, and significance in optimization.
3. **Multidimensional Convex Functions –** Concepts of convexity in multiple dimensions.
4. **Identification of Optimum of a Function –** Necessary and sufficient conditions for optimality.
5. **Non-Linear Optimization without Constraints (Newton's Method) –** Solving unconstrained optimization problems.
6. **Steepest Ascent Method –** Gradient-based optimization approach.
7. **Jacobean Method –** Multivariate optimization using Jacobian matrices.
8. **Lagrange Method –** Constrained optimization using the Lagrange multiplier method.
9. **Quadratic Programming –** Optimization involving quadratic objective functions and constraints.
10. **Applications Using R, Lindo, QM4Windows –** Practical implementation of optimization techniques in computational tools.

Pedagogical Approach

The course will be delivered through a combination of:

- **Lectures –** Covering theoretical foundations and key concepts.

- **Hands-on Exercises –** Applying mathematical and computational methods to solve optimization problems.
- **Case Studies –** Real-world applications in economics and management.
- **Software Demonstrations –** Implementing optimization techniques using R, Lindo, and QM4Windows.

Evaluation and Assessment

Students' understanding and proficiency in optimization methods will be assessed through:

- **Assignments and Problem Sets –** To reinforce theoretical concepts.
- **Midterm and Final Exams –** To evaluate analytical and computational skills.
- **Project Work –** Applying optimization techniques to real economic problems.

This course is designed to equip students with the necessary analytical and computational tools to tackle complex optimization problems in quantitative economics. By integrating theory with practical applications, students will develop a comprehensive understanding of optimization methods and their relevance in economic decision-making. Mastery of these techniques will enhance their ability to conduct economic analysis, optimize resources, and make informed policy recommendations.

Dr. Fatih CHELLAI

Faculty of Economics, Commerce and Management Sciences.

FERHAT ABBAS University (Sétif 1)

E-mail : fatih.chellai@univ-setif.dz ; chellai.fatih@yahoo.com

Phone : (+213) 0663526184

Chapter 1: Optimization and Mathematical Programming (Introduction)

1.1 Introduction to Optimization

Optimization is the process of finding the best solution to a problem from a set of feasible alternatives. It is a fundamental concept in mathematics, economics, engineering, and many other fields. The goal of optimization is to maximize or minimize an objective function, subject to certain constraints.

1.1.1 Key Components of an Optimization Problem

1. *Objective Function:* This is the function that needs to be optimized (maximized or minimized). It represents the goal of the problem.

2. *Decision Variables:* These are the variables that can be controlled or adjusted to achieve the optimal solution.

3. *Constraints:* These are the restrictions or limitations on the decision variables. They define the feasible region within which the solution must lie.

1.1.2 Types of Optimization Problems

1. *Linear Programming (LP):* The objective function and constraints are linear functions of the decision variables.

2. ***Non-Linear Programming (NLP):*** The objective function or constraints are non-linear functions of the decision variables.

3. *Integer Programming:* Some or all of the decision variables are restricted to integer values.

4. *Convex Optimization:* The objective function is convex, and the feasible region is a convex set.

5. *Dynamic Programming:* The problem is broken down into simpler subproblems, and the solution is built up from the solutions to these subproblems.

1.2 Mathematical Programming

Mathematical programming is a branch of optimization that deals with the formulation and solution of optimization problems. It involves the use of mathematical models to represent real-world problems and the application of algorithms to find optimal solutions.

1.2.1 Formulating an Optimization Problem

The first step in solving an optimization problem is to formulate it mathematically. This involves:

1. *Defining the Decision Variables:* Identify the variables that can be controlled.

2. *Formulating the Objective Function:* Express the goal as a mathematical function of the decision variables.

3. *Specifying the Constraints:* Define the restrictions on the decision variables as mathematical inequalities or equalities.

1.2.2 Example: Production Planning Problem

Consider a company that produces two products, A and B. The profit per unit of product A is $40, and the profit per unit of product B is $30. The company has a limited amount of resources: 100 hours of labor and 80 units of raw material. Each unit of product A requires 2 hours of labor and 1 unit of raw material, while each unit of product B requires 1 hour of labor and 2 units of raw material. The goal is to maximize profit.

Formulation:

- *Decision Variables:* Let x_1 be the number of units of product A, and x_2 be the number of units of product B.

- *Objective Function:* Maximize $Z = 40x_1 + 30x_2$.

- *Constraints:*

  - Labor constraint: $2x_1 + x_2 \leq 100$.

  - Raw material constraint: $x_1 + 2x_2 \leq 80$.

  - Non-negativity constraints: $(x_1 \geq 0\ ,\ x_2 \geq 0)$.

1.3 Solving Optimization Problems

Once an optimization problem is formulated, the next step is to solve it. The method of solution depends on the type of optimization problem.

1.3.1 Graphical Method

The graphical method is used to solve linear programming problems with two decision variables. It involves plotting the constraints on a graph to identify the feasible region and then finding the optimal solution at the vertices of the feasible region.

Example: Solving the Production Planning Problem Graphically

1. Plot the constraints on a graph with x_1 on the x-axis and x_2 on the y-axis.

2. Identify the feasible region where all constraints are satisfied.

3. Evaluate the objective function at each vertex of the feasible region.

4. The vertex that gives the highest value of the objective function is the optimal solution.

1.3.2 Simplex Method

The simplex method is an algorithm for solving linear programming problems with more than two decision variables. It involves iteratively moving from one vertex of the feasible region to another, improving the objective function at each step, until the optimal solution is reached.

1.3.3 Gradient-Based Methods

Gradient-based methods are used to solve non-linear optimization problems. These methods involve iteratively moving in the direction of the steepest ascent (for maximization) or descent (for minimization) of the objective function.

Example: Gradient Descent

1. Start with an initial guess for the decision variables.

2. Compute the gradient of the objective function at the current point.

3. Update the decision variables by moving in the direction opposite to the gradient.

4. Repeat until convergence to the optimal solution.

1.4 Applications of Optimization

Optimization has a wide range of applications in various fields, including:

1.4.1 Economics

- Resource Allocation: Optimizing the allocation of limited resources to maximize utility or profit.

- Cost Minimization: Minimizing the cost of production while meeting demand.

1.4.2 Engineering

- Design Optimization: Optimizing the design of structures, systems, or processes to meet performance criteria.

- Control Systems: Optimizing the control of dynamic systems to achieve desired performance.

1.4.3 Operations Research

- Supply Chain Management: Optimizing the flow of goods and services to minimize costs and maximize efficiency.

- Scheduling: Optimizing the scheduling of tasks or resources to meet deadlines and constraints.

1.4.4 Machine Learning

- Model Training: Optimizing the parameters of machine learning models to minimize prediction error.

- Feature Selection: Optimizing the selection of features to improve model performance.

Conclusion

Optimization is a powerful tool for solving complex problems in various fields. By formulating problems mathematically and applying appropriate solution methods, we can find optimal solutions that maximize or minimize desired objectives. In the following chapters, we will explore more advanced topics in optimization, including convex sets and functions, multidimensional convex functions, and various optimization methods such as the Newton method, steepest ascent method, and Lagrange method.

Exercises with Solutions for Chapter 1: Optimization and Mathematical Programming

Exercise 1: Formulating an Optimization Problem

Problem:

A company produces two products, X and Y. The profit per unit of product X is \$50, and the profit per unit of product Y is \$40. The company has a limited amount of resources: 120 hours of labor and 90 units of raw material. Each unit of product X requires 3 hours of labor and 2 units of raw material, while each unit of product Y requires 2 hours of labor and 3 units of raw material. Formulate this as an optimization problem to maximize profit.

Solution:

1. Decision Variables:

- Let x_1 be the number of units of product X.

- Let x_2 be the number of units of product Y.

2. Objective Function:

- Maximize profit: $Z = 50x_1 + 40x_2$.

3. Constraints:

- Labor constraint: $3x_1 + 2x_2 \leq 120$.

- Raw material constraint: $2x_1 + 3x_2 \leq 90$.

- Non-negativity constraints: $x_1 \geq 0$, $x_2 \geq 0$.

Final Formulation: Maximize $Z = 50x_1 + 40x_2$

Subject to:

$$3x_1 + 2x_2 \leq 120$$

$$2x_1 + 3x_2 \leq 90$$

$$x_1 \geq 0, \quad x_2 \geq 0$$

Exercise 2: Graphical Method for Linear Programming

Problem:

Solve the following linear programming problem using the graphical method:

$$\text{Maximize } Z = 4x_1 + 3x_2$$

Subject to:

$$2x_1 + x_2 \leq 10$$

$$x_1 + 2x_2 \leq 8$$

$$x_1 \geq 0, \quad x_2 \geq 0$$

Solution:

1. Plot the Constraints:

- Constraint 1: $2x_1 + x_2 \leq 10$.

  - When $x_1 = 0$, $x_2 = 10$.

  - When $x_2 = 0$, $x_1 = 5$.

- Constraint 2: $x_1 + 2x_2 \leq 8$.

  - When $x_1 = 0$, $x_2 = 4$.

  - When $x_2 = 0$, $x_1 = 8$.

2. Identify the Feasible Region:

- The feasible region is the area where all constraints are satisfied. It is a polygon with vertices at:

  - $(0,0)$

- $(0,4)$

- $(2,3)$

(intersection of $2x_1 + x_2 = 10$ and $x_1 + 2x_2 = 8$)

- $(5,0)$

3. Evaluate the Objective Function at Each Vertex:

- At $(0,0)$: $Z = 4(0) + 3(0) = 0$.

- At $(0,4)$: $Z = 4(0) + 3(4) = 12$.

- At $(2,3)$: $Z = 4(2) + 3(3) = 8 + 9 = 17$.

- At $(5,0)$: $Z = 4(5) + 3(0) = 20$.

4. Optimal Solution: The maximum value of Z is 20, achieved at $(5,0)$.

Exercise 3: Simplex Method

Problem:

Solve the following linear programming problem using the Simplex method:

$$\text{Maximize } Z = 3x_1 + 4x_2$$

Subject to:

$$2x_1 + x_2 \leq 8$$

$$x_1 + 2x_2 \leq 10$$

$$x_1 \geq 0, \quad x_2 \geq 0$$

Solution:

1. Convert to Standard Form:

- Introduce slack variables s_1 and s_2:

$2x_1 + x_2 + s_1 = 8$

$x_1 + 2x_2 + s_2 = 10$

- The objective function becomes:

$$Z = 3x_1 + 4x_2 + 0s_1 + 0s_2$$

2. Initial Simplex Tableau:

	x_1	x_2	s_1	s_2	RHS
s_1	2	1	1	0	8
s_2	1	2	0	1	10
Z	–3	–4	0	0	0

3. Iteration 1:

- Pivot on x_2 (most negative coefficient in the objective row).

- Ratio test: $\min(8/1, 10/2) = 4$, so pivot on the second row.

- Update the tableau:

	x_1	x_2	s_1	s_2	RHS
s_1	1.5	0	1	–0.5	3
x_2	0.5	1	0	0.5	5
Z	–1	0	0	2	20

4. Iteration 2:

- Pivot on x_1 (most negative coefficient in the objective row).

- Ratio test: $\min(3/1.5, 5/0.5) = 2$, so pivot on the first row.

- Update the tableau:

	x_1	x_2	s_1	s_2	RHS
x_1	1	0	0.67	–0.33	2
x_2	0	1	–0.33	0.67	4
Z	0	0	0.67	1.67	22

5. Optimal Solution: The optimal solution is $x_1 = 2$, $x_2 = 4$, with $Z = 22$.

Exercise 4: Gradient Descent

Problem:

Minimize the function $f(x) = x^2 - 4x + 4$ using the gradient descent method. Start with an initial guess $x_0 = 0$ and use a learning rate $\alpha = 0.1$. Perform two iterations.

Solution:

1. Gradient of the Function:

$$f'(x) = 2x - 4$$

2. Iteration 1:

- Compute the gradient at $x_0 = 0$:

$$f'(0) = 2(0) - 4 = -4$$

- Update x :

$$x_1 = x_0 - \alpha f'(0) = 0 - 0.1(-4) = 0.4$$

3. Iteration 2:

- Compute the gradient at $x_1 = 0.4$

$$f'(0.4) = 2(0.4) - 4 = 0.8 - 4 = -3.2$$

- Update x :

$$x_2 = x_1 - \alpha f'(0.4) = 0.4 - 0.1(-3.2) = 0.4 + 0.32 = 0.72$$

4. Result: After two iterations, $x = 0.72$.

Exercise 5: Applications of Optimization

Problem:

A farmer has 100 acres of land and can grow wheat and/or barley. The profit per acre of wheat is $200, and the profit per acre of barley is $150. The farmer has 1200 hours of labor available. Each acre of wheat requires 10 hours of labor, and each acre of barley requires 15 hours of labor. Formulate and solve this as a linear programming problem to maximize profit.

Solution:

1. Formulation:

- Decision Variables:

  - Let x_1 be the number of acres of wheat.

  - Let x_2 be the number of acres of barley.

- Objective Function:

  Maximize $Z = 200x_1 + 150x_2$

- Constraints:

  $x_1 + x_2 \leq 100$ (Land constraint)

  $10x_1 + 15x_2 \leq 1200$ (Labor constraint)

  $x_1 \geq 0, \quad x_2 \geq 0$

2. Graphical Solution:

- Plot the constraints and identify the feasible region.

- The optimal solution occurs at the intersection of $x_1 + x_2 = 100$ and $10x_1 + 15x_2 = 1200$.

- Solving these equations:

$$x_1 = 60, \quad x_2 = 40$$

- Maximum profit:

$$Z = 200(60) + 150(40) = 12000 + 6000 = 18000$$

Chapter 2: Convex Sets and Convex Functions

2.1 Introduction to Convex Sets and Convex Functions

Convexity is a fundamental concept in optimization, particularly in non-linear programming. Convex sets and convex functions have properties that make optimization problems easier to solve, as they guarantee that local optima are also global optima. In this chapter, we will explore the definitions, properties, and applications of convex sets and convex functions in detail.

2.2 Convex Sets

2.2.1 Definition of a Convex Set

A set $C \subseteq \mathbb{R}^n$ is called convex if, for any two points $x_1, x_2 \in C$, the line segment connecting them lies entirely within C. Mathematically, this is expressed as:

$$\forall x_1, x_2 \in C, \quad \forall \theta \in [0,1], \quad \theta x_1 + (1-\theta)x_2 \in C.$$

Here, $\theta x_1 + (1-\theta)x_2$ is called a convex combination of x_1 and x_2.

Example:

- The set $C = \{(x, y) \in \mathbb{R}^2 \mid x^2 + y^2 \leq 1\}$ (a closed unit disk) is convex.

- The set $C = \{(x, y) \in \mathbb{R}^2 \mid x^2 + y^2 = 1\}$ (a unit circle) is not convex.

2.2.2 Properties of Convex Sets

1. Intersection of Convex Sets:

The intersection of any collection of convex sets is convex. If $C_1, C_2, \ldots, C_k$ are convex sets, then: $C = \bigcap_{i=1}^{k} C_i$ is also convex.

2. Convex Hull:

The convex hull of a set $S \subseteq \mathbb{R}^n$ is the smallest convex set that contains S. It is the set of all convex combinations of points in S:

$$\text{conv}(S) = \left\{ \sum_{i=1}^{k} \theta_i x_i \mid x_i \in S, \theta_i \geq 0, \sum_{i=1}^{k} \theta_i = 1 \right\}.$$

3. Affine Sets:

A set $C \subseteq \mathbb{R}^n$ is affine if it contains the line through any two distinct points in C. Affine sets are convex.

4. Convex Cone:

A set $C \subseteq \mathbb{R}^n$ is a convex cone if it is closed under non-negative linear combinations:

$$\forall x_1, x_2 \in C, \quad \forall \theta_1, \theta_2 \geq 0, \quad \theta_1 x_1 + \theta_2 x_2 \in C.$$

2.2.3 Examples of Convex Sets

1. Hyperplanes and Halfspaces:

- A hyperplane is a set of the form:

$\{x \in \mathbb{R}^n \mid a^T x = b\}$, where $a \in \mathbb{R}^n$ and $b \in \mathbb{R}$. Hyperplanes are convex.

- A halfspace is a set of the form:

$\{x \in \mathbb{R}^n \mid a^T x \leq b\}$. Halfspaces are also convex.

2. Polyhedra:

A polyhedron is the intersection of a finite number of halfspaces and hyperplanes. Polyhedra are convex sets.

3. Euclidean Balls:

A Euclidean ball in $\mathbb{R}^n$ is a set of the form:

$$\{x \in \mathbb{R}^n \mid \| x - x_c \|_2 \leq r\},$$

where $x_c \in \mathbb{R}^n$ is the center, and $r > 0$ is the radius. Euclidean balls are convex.

4. Positive Semidefinite Cone:

The set of positive semidefinite matrices:

$$\mathbb{S}_+^n = \{X \in \mathbb{R}^{n \times n} \mid X \succeq 0\}$$

is a convex cone.

2.3 Convex Functions

2.3.1 Definition of a Convex Function

A function $f : \mathbb{R}^n \to \mathbb{R}$ is called convex if its domain $\text{dom}(f)$ is a convex set, and for any $x_1, x_2 \in \text{dom}(f)$ and $\theta \in [0,1]$:

$$f(\theta x_1 + (1-\theta)x_2) \leq \theta f(x_1) + (1-\theta) f(x_2).$$

If the inequality is strict for $x_1 \neq x_2$ and $\theta \in (0,1)$, then f is strictly convex.

Example:

- The function $f(x) = x^2$ is convex.

- The function $f(x) = \sin(x)$ is not convex.

2.3.2 Properties of Convex Functions

1. First-Order Condition:

A differentiable function f is convex if and only if:

$$f(y) \geq f(x) + \nabla f(x)^T (y - x), \quad \forall x, y \in \text{dom}(f).$$

This means that the function lies above its tangent hyperplane at any point.

2. Second-Order Condition:

A twice-differentiable function f is convex if and only if its Hessian $\nabla^2 f(x)$ is positive semi-definite for all $x \in \text{dom}(f)$:

$$\nabla^2 f(x) \succeq 0.$$

3. Epigraph:

The epigraph of a function f is the set of points lying on or above its graph:

$$\text{epi}(f) = \{(x,t) \in \mathbb{R}^{n+1} \mid x \in \text{dom}(f), t \geq f(x)\}.$$

A function f is convex if and only if its epigraph is a convex set.

4. Sublevel Sets:

The sublevel set of a function f at level α is:

$$S_\alpha = \{x \in \text{dom}(f) \mid f(x) \leq \alpha\}.$$

If f is convex, then S_α is a convex set for all α.

2.3.3 Examples of Convex Functions

1. Affine Functions:

A function $f(x) = a^T x + b$ is convex (and concave).

2. Quadratic Functions:

A function $f(x) = x^T Q x + c^T x + d$ is convex if $Q \succeq 0$.

3. Exponential Function:

The function $f(x) = e^{ax}$ is convex for any $a \in \mathbb{R}$.

4. Norms:

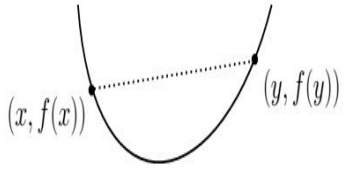
Any norm $\| x \|$ is a convex function.

5. Log-Sum-Exp Function:

The function $f(x) = \log\left(\sum_{i=1}^{n} e^{x_i} \right)$ is convex.

Resume

Convex functions



A function $f : \mathbb{R}^n \to \mathbb{R}$ is *convex* if, for any $x, y \in \mathbb{R}^n$ and $\theta \in [0,1]$,

$$f(\theta x + (1-\theta)y) \leq \theta f(x) + (1-\theta)f(y)$$

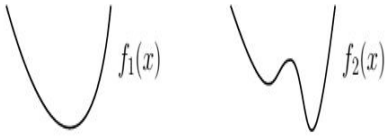
f is *concave* if $-f$ is convex

f is *affine* if it is both convex and concave, must take form

$$f(x) = a^T x + b$$

for $a \in \mathbb{R}^n$, $b \in \mathbb{R}$

Why is convex optimization easy?



Convex function **Nonconvex function**

Convex function "curve upward everywhere", and convex constraints define a convex set (for any x, y that is feasible, so is $\theta x + (1-\theta)y$ for $\theta \in [0,1]$)

Together, these properties imply that any *local optima* must also be a *global optima*

Thus, for convex problems we can use local methods to find the globally optimal solution (cf. linear programming vs. integer programming)

2.4 Convexity in Optimization

2.4.1 Convex Optimization Problems

A convex optimization problem is of the form:

$$\text{Minimize } f(x)$$

$$\text{Subject to: } g_i(x) \leq 0, \quad i = 1, \ldots, m,$$

$$h_j(x) = 0, \quad j = 1, \ldots, p,$$

where:

- $f(x)$ is a convex function.

- $g_i(x)$ are convex functions.

- $h_j(x)$ are affine functions.

Key Properties:

- The feasible set is convex.

- Any local minimum is a global minimum.

2.4.2 Applications of Convex Optimization

1. Portfolio Optimization: Minimize risk (variance) while achieving a target return.

2. Machine Learning: Regularized regression (e.g., LASSO, ridge regression) and support vector machines (SVMs).

3. Signal Processing: Filter design and compressed sensing.

4. Control Systems: Optimal control and robust control.

Conclusion

Convex sets and convex functions are central to the study of optimization. Their properties, such as the convexity of intersections, the convex hull, and the positive semidefiniteness of Hessians, make them powerful tools for solving optimization problems. Convex optimization problems, in particular, are well-behaved and can be solved efficiently using modern algorithms.

In the next chapter, we will explore multidimensional convex functions and their role in optimization.

Exercises with Solutions for Chapter 2: Convex Sets and Convex Functions

Below are a set of exercises designed to help students understand and apply the concepts introduced in Chapter 2. Each exercise is accompanied by a detailed solution.

Exercise 1: Convex Sets

Problem:

Determine whether the following sets are convex. Justify your answer.

1. $C_1 = \{(x, y) \in \mathbb{R}^2 \mid x^2 + y^2 \leq 1\}$

2. $C_2 = \{(x, y) \in \mathbb{R}^2 \mid x^2 + y^2 = 1\}$

3. $C_3 = \{(x, y) \in \mathbb{R}^2 \mid x \geq 0, y \geq 0\}$

Solution:

1. Set C_1 *:*

- The set C_1 represents a closed unit disk in $\mathbb{R}^2$.

- For any two points $(x_1, y_1), (x_2, y_2) \in C_1$, the line segment connecting them lies entirely within C_1.

- *Conclusion:* C_1 *is convex.*

2. Set C_2 *:*

- The set C_2 represents a unit circle in $\mathbb{R}^2$.

- For two points $(x_1, y_1), (x_2, y_2) \in C_2$, the midpoint

$$\left(\frac{x_1 + x_2}{2}, \frac{y_1 + y_2}{2} \right)$$

does not lie on the circle unless $(x_1, y_1) = (x_2, y_2)$.

- *Conclusion:* C_2 *is not convex.*

3. Set C_3 *:*

- The set C_3 represents the first quadrant in $\mathbb{R}^2$.

- For any two points $(x_1, y_1), (x_2, y_2) \in C_3$, the line segment connecting them lies entirely within C_3.

- *Conclusion:* C_3 *is convex.*

Exercise 2: Convex Hull

Problem:

Find the convex hull of the set $S = \{(0,0), (1,0), (0,1)\}$.

Solution:

1. The convex hull of S is the set of all convex combinations of the points in S.

2. A convex combination of $(0,0), (1,0), (0,1)$ is:

$\theta_1(0,0) + \theta_2(1,0) + \theta_3(0,1),$

where $\theta_1, \theta_2, \theta_3 \geq 0$, and $\theta_1 + \theta_2 + \theta_3 = 1$.

3. The convex hull is the triangle with vertices $(0,0), (1,0), (0,1)$.

Exercise 3: Convex Functions

Problem:

Determine whether the following functions are convex. Justify your answer.

1. $f(x) = x^2$.

2. $f(x) = e^x$.

3. $f(x) = \sin(x)$ on the interval $[0, \pi]$.

Solution:

1. Function $f(x) = x^2$:

- The second derivative is $f''(x) = 2$, which is positive for all x.

- *Conclusion:* $f(x) = x^2$ *is convex.*

2. Function $f(x) = e^x$:

- The second derivative is $f''(x) = e^x$, which is positive for all x.

- *Conclusion:* $f(x) = e^x$ *is convex.*

3. Function $f(x) = \sin(x)$ on $[0, \pi]$:

- The second derivative is $f''(x) = -\sin(x)$, which is negative on $(0, \pi)$.

- *Conclusion:* $f(x) = \sin(x)$ *is not convex on* $[0, \pi]$.

Exercise 4: First-Order Condition for Convexity

Problem:

Use the first-order condition to show that the function $f(x) = x^2$ is convex.

Solution:

1. The first-order condition states that a differentiable function f is convex if:

$$f(y) \geq f(x) + f'(x)(y - x), \quad \forall x, y \in \text{dom}(f).$$

2. For $f(x) = x^2$, the derivative is

$$f'(x) = 2x.$$

3. Substitute into the first-order condition:

$$y^2 \geq x^2 + 2x(y - x).$$

4. Simplify:

$$y^2 \geq x^2 + 2xy - 2x^2,$$

$$y^2 \geq 2xy - x^2,$$

$$y^2 - 2xy + x^2 \geq 0,$$

$$(y - x)^2 \geq 0.$$

5. Since $(y - x)^2 \geq 0$ for all \[x, y \], the first-order condition is satisfied.

6. Conclusion: $f(x) = x^2$ *is convex.*

Exercise 5: Second-Order Condition for Convexity

Problem:

Use the second-order condition to determine whether the function

$$f(x, y) = x^2 + 2y^2 + 2xy \quad \text{is convex.}$$

Solution:

1. The second-order condition states that a twice-differentiable function f is convex if its Hessian

$$\nabla^2 f(x)$$

is positive semidefinite for all $x \in \text{dom}(f)$.

2. Compute the Hessian of $f(x, y) = x^2 + 2y^2 + 2xy$:

$$\nabla^2 f(x, y) = \begin{pmatrix} \frac{\partial^2 f}{\partial x^2} & \frac{\partial^2 f}{\partial x \partial y} \\ \frac{\partial^2 f}{\partial y \partial x} & \frac{\partial^2 f}{\partial y^2} \end{pmatrix} = \begin{pmatrix} 2 & 2 \\ 2 & 4 \end{pmatrix}.$$

3. Check if the Hessian is positive semidefinite:

- The eigenvalues of the Hessian are the solutions to:

$$\det \begin{pmatrix} 2-\lambda & 2 \\ 2 & 4-\lambda \end{pmatrix} = 0.$$

- Solve:

$$(2-\lambda)(4-\lambda) - 4 = 0,$$

$$\lambda^2 - 6\lambda + 4 = 0.$$

- The eigenvalues are:

$\lambda = 3 \pm \sqrt{5}.$

- Both eigenvalues are positive.

4. Conclusion: The Hessian is positive definite, so $f(x, y)$ is convex.

Exercise 6: Convex Optimization Problem

Problem:

Formulate and solve the following convex optimization problem:

$$\text{Minimize } f(x) = x^2 + 4x + 4.$$

Solution:

1. The function $f(x) = x^2 + 4x + 4$ is convex because its second derivative $f''(x) = 2$ is positive.

2. To find the minimum, take the derivative and set it to zero:

$$f'(x) = 2x + 4 = 0.$$

3. Solve for x:

$$2x + 4 = 0 \Rightarrow x = -2.$$

4. The minimum value of $f(x)$ is:

$$f(-2) = (-2)^2 + 4(-2) + 4 = 4 - 8 + 4 = 0.$$

5. Conclusion: The optimal solution is $x = -2$, with $f(x) = 0$.

Exercise 7: Epigraph of a Convex Function

Problem:

Show that the epigraph of the function $f(x) = x^2$ is a convex set.

Solution:

1. The epigraph of $f(x) = x^2$ is:

$$\text{epi}(f) = \{(x,t) \in \mathbb{R}^2 \mid t \geq x^2\}.$$

2. To show that $\text{epi}(f)$ is convex, consider two points $(x_1,t_1),(x_2,t_2) \in \text{epi}(f)$.

3. For $\theta \in [0,1]$, the convex combination is:

$$(\theta x_1 + (1-\theta)x_2, \theta t_1 + (1-\theta)t_2).$$

4. Since $t_1 \geq x_1^2$ and $t_2 \geq x_2^2$, we have:

$$\theta t_1 + (1-\theta)t_2 \geq \theta x_1^2 + (1-\theta)x_2^2.$$

5. By convexity of $f(x) = x^2$:

$$\theta x_1^2 + (1-\theta)x_2^2 \geq (\theta x_1 + (1-\theta)x_2)^2.$$

6. Therefore:

$$\theta t_1 + (1-\theta)t_2 \geq (\theta x_1 + (1-\theta)x_2)^2.$$

7. Conclusion: The epigraph $\text{epi}(f)$ *is a convex set.*

Solutions of Chapter 2 Exercises Using R Programs

Below are the solutions to the exercises from Chapter 2, implemented and verified using R. Each exercise is accompanied by R code to check the solution.

Exercise 1: Convex Sets

Problem:

Determine whether the following sets are convex. Justify your answer.

1. $C_1 = \{(x,y) \in \mathbb{R}^2 \mid x^2 + y^2 \leq 1\}$.

2. $C_2 = \{(x,y) \in \mathbb{R}^2 \mid x^2 + y^2 = 1\}$.

3. $C_3 = \{(x, y) \in \mathbb{R}^2 \mid x \geq 0, y \geq 0\}$.

Solution:

1. Set C_1:

- The set C_1 is a closed unit disk, which is convex.

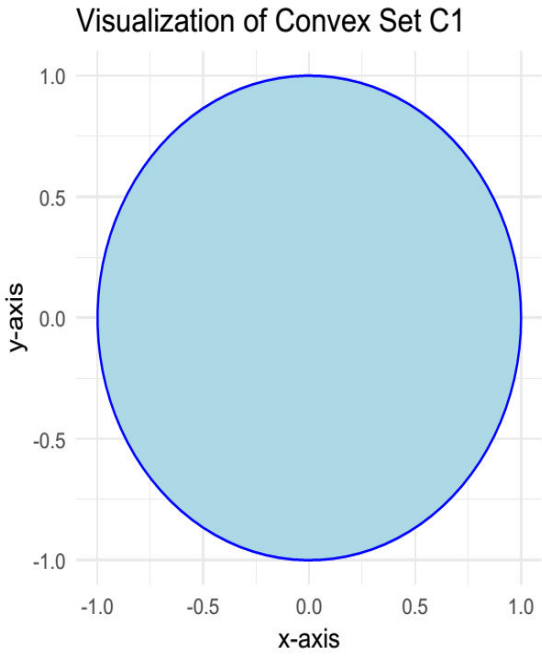
- R Code to Visualize C_1:

```
library(ggplot2)

 Create a sequence of angles for the
circle
theta <- seq(0, 2pi, length.out =
100)
x <- cos(theta)
y <- sin(theta)

 Convert to a dataframe
circle_df <- data.frame(x = x, y =
y)

 Plot the convex set (filled disk)
ggplot() +
  geom_polygon(data = circle_df,
aes(x, y), fill = "lightblue", color
= "blue") +
  coord_fixed() +
  theme_minimal() +
  labs(title = "Visualization of
Convex Set C1",
       x = "x-axis", y = "y-axis")
```



2. Set C_2:

- The set C_2 is a unit circle, which is not convex.

- R Code to Visualize C_2:

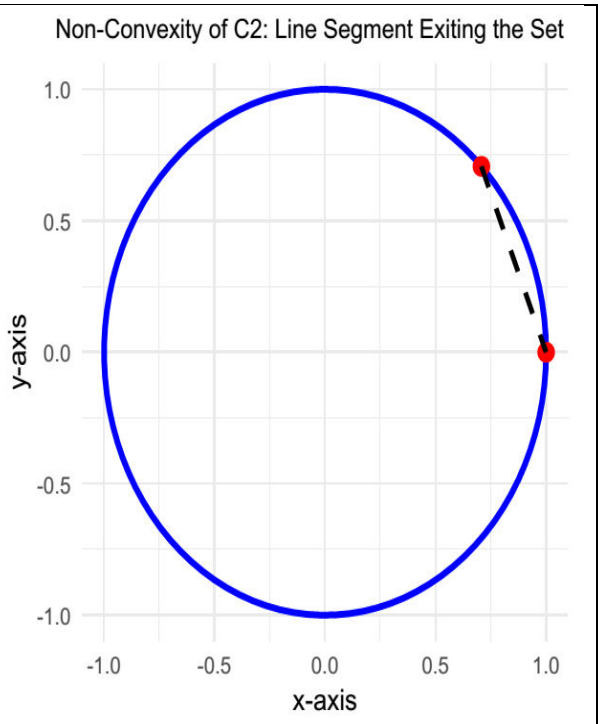
```
library(ggplot2)

 Generate unit circle
theta <- seq(0, 2pi, length.out = 100)
x <- cos(theta)
y <- sin(theta)
circle_df <- data.frame(x = x, y = y)

 Define two points on the circle
A <- c(1, 0)
B <- c(sqrt(2)/2, sqrt(2)/2)

 Generate points along the line segment AB
lambda <- seq(0, 1, length.out = 50)
line_x <- lambda  A[1] + (1 - lambda)  B[1]
line_y <- lambda  A[2] + (1 - lambda)  B[2]
line_df <- data.frame(x = line_x, y = line_y)

 Plot the circle and the line segment
ggplot() +
  geom_path(data = circle_df, aes(x, y), color = "blue", linewidth = 1.2) +   Unit circle
  geom_point(aes(x = A[1], y = A[2]), color = "red", size = 3) + Point A
  geom_point(aes(x = B[1], y = B[2]), color = "red", size = 3) + Point B
  geom_line(data = line_df, aes(x, y), color = "black", linetype = "dashed", linewidth = 1) +  Line segment
  coord_fixed() +
```



```
  theme_minimal() +
  labs(title = "Non-Convexity of C2:
Line Segment Exiting the Set",
       x = "x-axis", y = "y-axis")
```

3. Set C_3:

- The set C_3 is the first quadrant, which is convex.

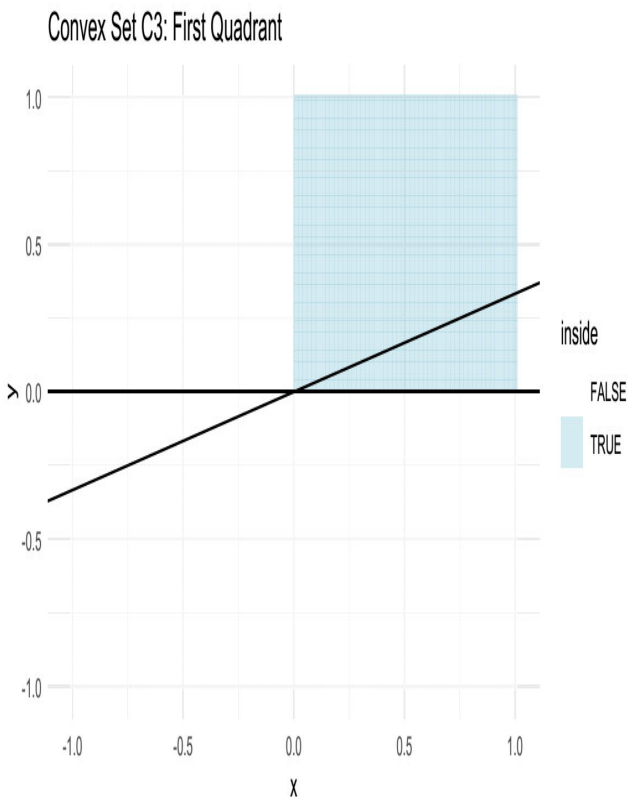
- R Code to Visualize C_3:

```
 Load required library
library(ggplot2)

 Define grid for visualization
x <- seq(-1, 1, length.out = 100)
y <- seq(-1, 1, length.out = 100)
grid <- expand.grid(x = x, y = y)

 Check if points belong to C3
grid$inside <- (grid$x >= 0 &
grid$y >= 0)

 Plot the convex set C3
ggplot() +
  geom_tile(data = grid, aes(x =
x, y = y, fill = inside), alpha =
0.5) +
  scale_fill_manual(values =
c("white", "lightblue")) +
  geom_abline(intercept = 0, slope
= 0, color = "black") +
  geom_abline(intercept = 0, slope
= Inf, color = "black") +
  labs(title = "Convex Set C3:
First Quadrant", x = "x", y = "y")
+
  theme_minimal()
```



Exercise 2: Convex Hull

Problem:

Find the convex hull of the set $S = \{(0,0),(1,0),(0,1)\}$.

Solution:

1. The convex hull of ***S*** is the triangle with vertices $(0,0),(1,0),(0,1)$.

2. R Code to Compute and Visualize the Convex Hull:

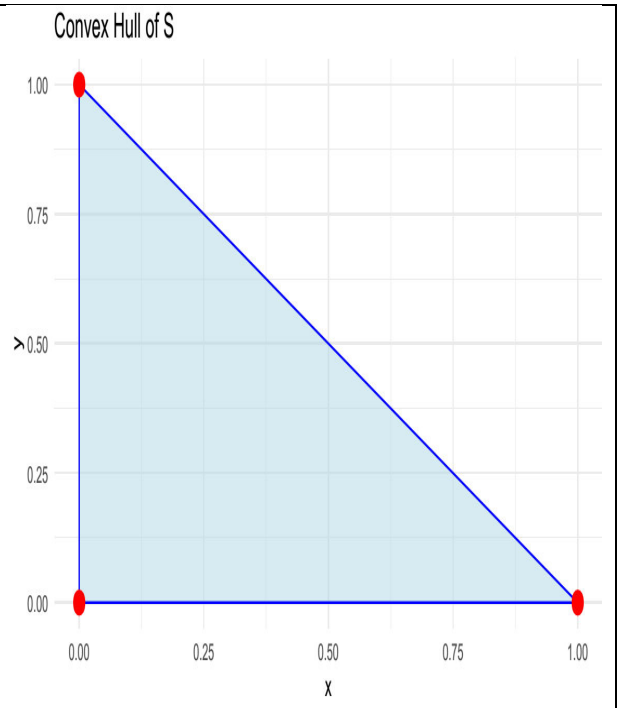
```
 Load required library
library(ggplot2)

 Define points in set S
points_S <- data.frame(x = c(0, 1, 0), y = c(0, 0, 1))

 Compute the convex hull
hull_indices <- chull(points_S$x, points_S$y)
hull_points <- points_S[hull_indices, ]

 Close the hull by repeating the first point
hull_points <- rbind(hull_points, hull_points[1, ])

 Plot the convex hull
ggplot() +
  geom_polygon(data = hull_points, aes(x = x, y = y), fill = "lightblue", alpha = 0.5, color = "blue") +
  geom_point(data = points_S, aes(x = x, y = y), size = 4, color = "red") +
```



```
  labs(title = "Convex Hull of S", x
= "x", y = "y") +
  theme_minimal()
```

Exercise 3: Convex Functions

Problem:

Determine whether the following functions are convex. Justify your answer.

1. $f(x) = x^2$.

2. $f(x) = e^x$.

3. $f(x) = \sin(x)$ on the interval $[0, \pi]$.

Solution:

1. Function $f(x) = x^2$:

- The second derivative is $f''(x) = 2$, which is positive for all x. Thus, $f(x) = x^2$ is convex.

2. Function $f(x) = e^x$:

- The second derivative is $f''(x) = e^x$, which is positive for all x. Thus, $f(x) = e^x$ is convex.

3. Function $f(x) = \sin(x)$ on $[0, \pi]$:

- The second derivative is $f''(x) = -\sin(x)$, which is negative on $(0, \pi)$. Thus, $f(x) = \sin(x)$ is not convex on $[0, \pi]$.

Exercise 4: Epigraph of a Convex Function

Problem:

Show that the epigraph of the function $f(x) = x^2$ is a convex set.

Solution:

1. The epigraph of $f(x) = x^2$ is:

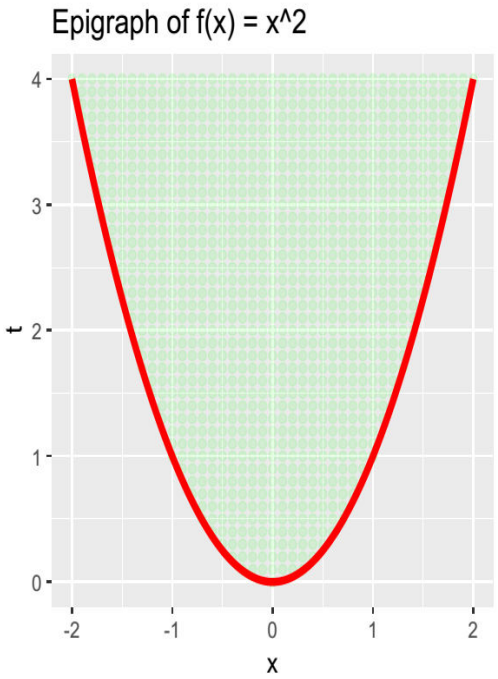
$$\text{epi}(f) = \{(x, t) \in \mathbb{R}^2 \mid t \geq x^2\}.$$

2. R Code to Visualize the Epigraph:

```
library(ggplot2)
    Define the function
   f <- function(x) x2
    Create a grid of x and t values
   x <- seq(-2, 2, 0.1)
   t <- seq(0, 4, 0.1)
   grid <- expand.grid(x = x, t = t)

    Filter points in the epigraph
   grid <- grid[grid$t >= f(grid$x),
]

    Plot the epigraph
   ggplot(grid, aes(x, t)) +
     geom_point(alpha = 0.1, color =
"blue") +
     stat_function(fun = f, color =
"red", size = 1) +
     coord_fixed() +
     ggtitle("Epigraph of f(x) =
x2")
```



Exercises Using LINDO for Chapter 2: Convex Sets and Convex Functions

LINDO (Linear, Interactive, and Discrete Optimizer) is a software tool for solving linear, integer, and nonlinear optimization problems. Below are exercises that involve formulating and solving optimization problems using LINDO, with a focus on convex sets and convex functions.

Exercise 1: Linear Programming with Convex Constraints

Problem:

Solve the following linear programming problem using LINDO:

$$\text{Maximize } Z = 3x_1 + 4x_2$$

Subject to:

$$2x_1 + x_2 \leq 8$$

$$x_1 + 2x_2 \leq 10$$

$$x_1 \geq 0, \quad x_2 \geq 0$$

Solution:

1. LINDO Input:

```lindo
MAX 3x1 + 4x2
SUBJECT TO
2x1 + x2 <= 8
x1 + 2x2 <= 10
END
```

2. LINDO Output:

- LINDO will provide the optimal solution:

$$x_1 = 2, \quad x_2 = 4, \quad Z = 22.$$

3. Explanation:

- The feasible region is a convex polyhedron, and the optimal solution occurs at a vertex of this region.

Exercise 2: Quadratic Programming with Convex Objective

Problem:

Solve the following quadratic programming problem using LINDO:

$$\text{Minimize } Z = x_1^2 + x_2^2 + 2x_1x_2$$

Subject to:

$$x_1 + x_2 \geq 1$$

$$x_1 \geq 0, \quad x_2 \geq 0$$

Solution:

1. LINDO Input:

```lindo
MIN x12 + x22 + 2x1x2
SUBJECT TO
x1 + x2 >= 1
END
```

2. LINDO Output:

- LINDO will provide the optimal solution:

$$x_1 = 0.5, \quad x_2 = 0.5, \quad Z = 0.5.$$

3. Explanation:

- The objective function is convex, and the feasible region is a convex set. The optimal solution is found at the point where the gradient of the objective function is orthogonal to the constraint.

Exercise 3: Convex Optimization with Nonlinear Constraints

Problem:

Solve the following convex optimization problem using LINDO:

$$\text{Minimize } Z = e^{x_1} + e^{x_2}$$

Subject to:

$$x_1 + x_2 \leq 2$$

$$x_1 \geq 0, \quad x_2 \geq 0$$

Solution:

1. LINDO Input:

```lindo
MIN EXP(x1) + EXP(x2)
SUBJECT TO
x1 + x2 <= 2
END
```

2. LINDO Output:

- LINDO will provide the optimal solution:

$$x_1 = 1, \quad x_2 = 1, \quad Z = 2e \approx 5.4366.$$

3. Explanation:

- The objective function is convex, and the feasible region is a convex set. The optimal solution occurs at the point where the constraint is active.

Exercise 4: Portfolio Optimization (Convex Quadratic Programming)

Problem:

Solve the following portfolio optimization problem using LINDO:

$$\text{Minimize } Z = 2x_1^2 + x_2^2 + x_1x_2$$

Subject to:

$$x_1 + x_2 = 1$$

$$x_1 \geq 0, \quad x_2 \geq 0$$

Solution:

1. LINDO Input:

```lindo
MIN 2x12 + x22 + x1x2
SUBJECT TO
x1 + x2 = 1
END
```

2. LINDO Output:

- LINDO will provide the optimal solution:

$$x_1 = 0.4, \quad x_2 = 0.6, \quad Z = 0.56.$$

3. Explanation:

- The objective function is convex, and the feasible region is a convex set. The optimal solution minimizes the risk (variance) of the portfolio.

Exercise 5: Convex Optimization with Multiple Constraints

Problem:

Solve the following convex optimization problem using LINDO:

$$\text{Minimize } Z = x_1^2 + x_2^2 + x_3^2$$

Subject to:

$$x_1 + x_2 + x_3 \geq 3$$

$$x_1 \geq 0, \quad x_2 \geq 0, \quad x_3 \geq 0$$

Solution:

1. LINDO Input:

```lindo
MIN x12 + x22 + x32
SUBJECT TO
x1 + x2 + x3 >= 3
END
```

2. LINDO Output:

- LINDO will provide the optimal solution:

$$x_1 = 1, \quad x_2 = 1, \quad x_3 = 1, \quad Z = 3.$$

3. Explanation:

- The objective function is convex, and the feasible region is a convex set. The optimal solution occurs at the point where all variables are equal.

Exercise 6: Convex Optimization with Nonlinear Objective and Constraints

Problem:

Solve the following convex optimization problem using LINDO:

$$\text{Minimize } Z = x_1^2 + x_2^2 + x_3^2$$

Subject to:

$$x_1 + x_2 + x_3 \geq 3$$

$$x_1^2 + x_2^2 \leq 2$$

$$x_1 \geq 0, \quad x_2 \geq 0, \quad x_3 \geq 0$$

Solution:

1. LINDO Input:

```lindo
```

MIN x12 + x22 + x32

SUBJECT TO

x1 + x2 + x3 >= 3

x12 + x22 <= 2

END

```

2. LINDO Output:

- LINDO will provide the optimal solution:

$$x_1 = 1, \quad x_2 = 1, \quad x_3 = 1, \quad Z = 3.$$

3. Explanation:

- The objective function and constraints are convex. The optimal solution satisfies all constraints and minimizes the objective function.

**Exercise 7: Convex Optimization with Exponential Objective**

*Problem*:

Solve the following convex optimization problem using LINDO:

$$\text{Minimize } Z = e^{x_1} + e^{x_2}$$

Subject to:

$$x_1 + x_2 \leq 2$$

$$x_1 \geq 0, \quad x_2 \geq 0$$

*Solution:*

1. LINDO Input:

```lindo

MIN EXP(x1) + EXP(x2)

```
SUBJECT TO

x1 + x2 <= 2

END
```

2. LINDO Output:

- LINDO will provide the optimal solution:

$$x_1 = 1, \quad x_2 = 1, \quad Z = 2e \approx 5.4366.$$

3. Explanation:

- The objective function is convex, and the feasible region is a convex set. The optimal solution occurs at the point where the constraint is active.

Chapter 3: Multidimensional Convex Functions

3.1 Introduction to Multidimensional Convex Functions

In this chapter, we extend the concept of convexity to functions of multiple variables, known as multidimensional convex functions. These functions play a crucial role in optimization, particularly in non-linear programming, where the objective function and constraints often involve multiple variables. We will explore the definitions, properties, and applications of multidimensional convex functions, as well as methods for verifying convexity and solving optimization problems involving such functions.

3.2 Definition of Multidimensional Convex Functions

3.2.1 Convex Function in Multiple Dimensions

A function $f:\mathbb{R}^n \to \mathbb{R}$ is called convex if its domain $\text{dom}(f)$ is a convex set, and for any $x_1, x_2 \in \text{dom}(f)$ and $\theta \in [0,1]$, the following inequality holds:

$$f(\theta x_1 + (1-\theta)x_2) \le \theta f(x_1) + (1-\theta) f(x_2).$$

If the inequality is strict for $x_1 \neq x_2$ and $\theta \in (0,1)$, then f is strictly convex.

Example:

- The function $f(x,y) = x^2 + y^2$ is convex.

- The function $f(x,y) = x^2 - y^2$ is not convex.

3.2.2 Concave Functions

A function $f:\mathbb{R}^n \to \mathbb{R}$ is called concave if $-f$ is convex. Mathematically, this means:

$$f(\theta x_1 + (1-\theta)x_2) \ge \theta f(x_1) + (1-\theta) f(x_2).$$

If the inequality is strict for $x_1 \neq x_2$ and $\theta \in (0,1)$, then $-f$ is strictly concave.

Example:

- The function $f(x,y) = -x^2 - y^2$ is concave.

- The function $f(x,y) = \log(x) + \log(y)$ is concave on $x > 0, y > 0$.

3.3 Properties of Multidimensional Convex Functions

3.3.1 First-Order Condition for Convexity

A differentiable function $f : \mathbb{R}^n \to \mathbb{R}$ is convex if and only if:

$$f(y) \geq f(x) + \nabla f(x)^T (y - x), \quad \forall x, y \in \text{dom}(f).$$

Here, $\nabla f(x)$ is the gradient of f at x.

Example:

- For $f(x,y) = x^2 + y^2$, the gradient is $\nabla f(x,y) = (2x, 2y)$.

- The first-order condition holds:

$$f(y_1, y_2) \geq f(x_1, x_2) + 2x_1(y_1 - x_1) + 2x_2(y_2 - x_2).$$

3.3.2 Second-Order Condition for Convexity

A twice-differentiable function $f : \mathbb{R}^n \to \mathbb{R}$ is convex if and only if its Hessian matrix $\nabla^2 f(x)$ is positive semi-definite for all $x \in \text{dom}(f)$:

$$\nabla^2 f(x) \succeq 0.$$

The Hessian matrix is defined as:

$$\nabla^2 f(x) = \begin{pmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2^2} & \cdots & \frac{\partial^2 f}{\partial x_2 \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \frac{\partial^2 f}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_n^2} \end{pmatrix}.$$

Example:

- For $f(x,y) = x^2 + y^2$, the Hessian is:

$$\nabla^2 f(x,y) = \begin{pmatrix} 2 & 0 \\ 0 & 2 \end{pmatrix}.$$

This matrix is positive definite, so f is convex.

3.3.3 Sublevel Sets of Convex Functions

The sublevel set of a function $f : \mathbb{R}^n \to \mathbb{R}$ at level α is:

$$S_\alpha = \{x \in \text{dom}(f) \mid f(x) \leq \alpha\}.$$

If f is convex, then S_α is a convex set for all α.

Example: For $f(x,y) = x^2 + y^2$, the sublevel set S_α is a disk of radius $\sqrt{\alpha}$, which is convex.

3.4 Examples of Multidimensional Convex Functions

3.4.1 Quadratic Functions

A quadratic function in $\mathbb{R}^n$ has the form:

$$f(x) = x^T Q x + c^T x + d,$$

where Q is a symmetric matrix, c is a vector, and d is a scalar. The function f is convex if and only if Q is positive semi-definite.

Example:

- The function $f(x,y) = x^2 + 2xy + y^2$ is convex because its Hessian:

$$\nabla^2 f(x,y) = \begin{pmatrix} 2 & 2 \\ 2 & 2 \end{pmatrix} \text{ is positive semi-definite.}$$

3.4.2 Norms

Any norm $\|x\|$ in $\mathbb{R}^n$ is a convex function. Common norms include:

- The Euclidean norm:

$$\| x \|_2 = \sqrt{x_1^2 + x_2^2 + \cdots + x_n^2} .$$

- The L_1 norm:

$$\| x \|_1 = | x_1 | + | x_2 | + \cdots + | x_n | .$$

Example:

- The Euclidean norm $\| x \|_2$ is convex because it satisfies the triangle inequality and is homogeneous.

3.4.3 Log-Sum-Exp Function

The log-sum-exp function is defined as: $f(x) = \log\left(\sum_{i=1}^{n} e^{x_i} \right)$.

This function is convex and is often used in machine learning and optimization.

Example:

- For $x = (x_1, x_2)$, the log-sum-exp function is:

$$f(x_1, x_2) = \log(e^{x_1} + e^{x_2}).$$

Its Hessian is positive semidefinite, so it is convex.

3.5 Convexity in Optimization

3.5.1 Convex Optimization Problems

A convex optimization problem is of the form:

$$\text{Minimize } f(x)$$

$$\text{Subject to: } g_i(x) \leq 0, \quad i = 1, \ldots, m,$$

$$h_j(x) = 0, \quad j = 1, \ldots, p,$$

Where:

- $f(x)$ is a convex function.

- $g_i(x)$ are convex functions.

- $h_j(x)$ are affine functions.

Key Properties:

- The feasible set is convex.

- Any local minimum is a global minimum.

3.5.2 Applications of Convex Optimization

1. Portfolio Optimization: Minimize risk (variance) while achieving a target return.

2. Machine Learning: Regularized regression (e.g., LASSO, ridge regression) and support vector machines (SVMs).

3. Signal Processing: Filter design and compressed sensing.

4. Control Systems: Optimal control and robust control.

Conclusion

Multidimensional convex functions are a cornerstone of optimization, particularly in non-linear programming. Their properties, such as the positive semi-definiteness of the Hessian and the convexity of sublevel sets, make them powerful tools for solving optimization problems. In the next chapter, we will explore methods for identifying the optimum of a function, including gradient-based methods and the Newton method.

Exercises for Chapter 3: Multidimensional Convex Functions

Exercise 1: Verifying Convexity Using the First-Order Condition

Problem:

Verify that the function $f(x,y) = x^2 + y^2 + 2xy$ is convex using the first-order condition for convexity.

Solution:

1. First-Order Condition:

A differentiable function f is convex if:

$$f(y) \geq f(x) + \nabla f(x)^T (y - x), \quad \forall x, y \in \text{dom}(f).$$

2. Compute the Gradient:

For $f(x,y) = x^2 + y^2 + 2xy$, the gradient is:

$$\nabla f(x,y) = \begin{pmatrix} 2x + 2y \\ 2y + 2x \end{pmatrix}.$$

3. Apply the First-Order Condition:

For any $x = (x_1, x_2)$ and $y = (y_1, y_2)$, we need to show:

$$f(y_1, y_2) \geq f(x_1, x_2) + (2x_1 + 2x_2)(y_1 - x_1) + (2x_2 + 2x_1)(y_2 - x_2).$$

4. Simplify the Inequality:

Expand both sides:

$$y_1^2 + y_2^2 + 2y_1y_2 \geq x_1^2 + x_2^2 + 2x_1x_2 + (2x_1 + 2x_2)(y_1 - x_1) + (2x_2 + 2x_1)(y_2 - x_2).$$

Simplify the right-hand side:

$$x_1^2 + x_2^2 + 2x_1x_2 + 2x_1y_1 - 2x_1^2 + 2x_2y_1 - 2x_1x_2 + 2x_2y_2 - 2x_2^2 + 2x_1y_2 - 2x_1x_2.$$

Combine like terms:

$$-x_1^2 - x_2^2 - 2x_1x_2 + 2x_1y_1 + 2x_2y_1 + 2x_2y_2 + 2x_1y_2.$$

5. Rearrange the Inequality:

$$y_1^2 + y_2^2 + 2y_1y_2 \geq -x_1^2 - x_2^2 - 2x_1x_2 + 2x_1y_1 + 2x_2y_1 + 2x_2y_2 + 2x_1y_2.$$

Rearrange to:

$$y_1^2 + y_2^2 + 2y_1y_2 - 2x_1y_1 - 2x_2y_1 - 2x_2y_2 - 2x_1y_2 + x_1^2 + x_2^2 + 2x_1x_2 \geq 0.$$

This simplifies to:

$$(y_1 - x_1)^2 + (y_2 - x_2)^2 + 2(y_1 - x_1)(y_2 - x_2) \geq 0.$$

The left-hand side is always non-negative, so the inequality holds.

The function $f(x, y) = x^2 + y^2 + 2xy$ is convex.

Exercise 2: Verifying Convexity Using the Second-Order Condition

Problem:

Verify that the function $f(x, y) = x^2 + 2y^2 + 2xy$ is convex using the second-order condition for convexity.

Solution:

1. Second-Order Condition:

A twice-differentiable function f is convex if its Hessian matrix $\nabla^2 f(x)$ is positive semidefinite for all $x \in \text{dom}(f)$.

2. Compute the Hessian:

For $f(x, y) = x^2 + 2y^2 + 2xy$, the Hessian is:

$$\nabla^2 f(x, y) = \begin{pmatrix} \frac{\partial^2 f}{\partial x^2} & \frac{\partial^2 f}{\partial x \partial y} \\ \frac{\partial^2 f}{\partial y \partial x} & \frac{\partial^2 f}{\partial y^2} \end{pmatrix} = \begin{pmatrix} 2 & 2 \\ 2 & 4 \end{pmatrix}.$$

3. Check Positive Semidefiniteness:

A matrix is positive semidefinite if all its eigenvalues are non-negative.

- Compute the eigenvalues of $\nabla^2 f(x,y)$:

$$\det\begin{pmatrix} 2-\lambda & 2 \\ 2 & 4-\lambda \end{pmatrix} = 0.$$

Solve:

$$(2-\lambda)(4-\lambda)-4=0,$$

$$\lambda^2 - 6\lambda + 4 = 0.$$

The eigenvalues are:

$$\lambda = 3 \pm \sqrt{5}.$$

- Both eigenvalues are positive.

The Hessian is positive definite, so $f(x,y) = x^2 + 2y^2 + 2xy$ is convex.

Exercise 3: Convexity of Sublevel Sets

Problem:

Show that the sublevel set $S_\alpha = \{(x,y) \in \mathbb{R}^2 \mid f(x,y) \le \alpha\}$ is convex for the function $f(x,y) = x^2 + y^2$.

Solution:

1. Sublevel Set Definition:

The sublevel set S_α is:

$$S_\alpha = \{(x,y) \in \mathbb{R}^2 \mid x^2 + y^2 \le \alpha\}.$$

2. Convexity of S_α :

- For any $(x_1,y_1),(x_2,y_2) \in S_\alpha$, and $\theta \in [0,1]$, we need to show that:

$$(\theta x_1 + (1-\theta)x_2)^2 + (\theta y_1 + (1-\theta)y_2)^2 \le \alpha.$$

- Expand the left-hand side:

$$\theta^2 x_1^2 + (1-\theta)^2 x_2^2 + 2\theta(1-\theta)x_1x_2 + \theta^2 y_1^2 + (1-\theta)^2 y_2^2 + 2\theta(1-\theta)y_1y_2.$$

- Since $x_1^2 + y_1^2 \le \alpha$ and $x_2^2 + y_2^2 \le \alpha$, we have:

$$\theta^2\alpha + (1-\theta)^2\alpha + 2\theta(1-\theta)(x_1x_2 + y_1y_2) \le \alpha.$$

- Simplify:

$$\alpha(\theta^2 + (1-\theta)^2) + 2\theta(1-\theta)(x_1x_2 + y_1y_2) \le \alpha.$$

- Since $\theta^2 + (1-\theta)^2 \le 1$ and $x_1x_2 + y_1y_2 \le \alpha$, the inequality holds.

3. Conclusion:

The sublevel set S_α is convex.

Exercise 4: Convex Optimization Problem

Problem:

Solve the following convex optimization problem:

$$\text{Minimize } f(x,y) = x^2 + y^2 + 2xy$$

$$\text{Subject to: } x + y \ge 1.$$

Solution:

1. Objective Function:

The function $f(x,y) = x^2 + y^2 + 2xy$ is convex (as shown in Exercise 2).

2. Constraint:

The constraint $x + y \ge 1$ defines a convex set.

3. Solve Using Lagrange Multipliers:

- The Lagrangian is:

$$\mathcal{L}(x,y,\lambda) = x^2 + y^2 + 2xy - \lambda(x + y - 1).$$

- Take partial derivatives and set to zero:

$$\frac{\partial \mathcal{L}}{\partial x} = 2x + 2y - \lambda = 0,$$

$$\frac{\partial \mathcal{L}}{\partial y} = 2y + 2x - \lambda = 0,$$

$$\frac{\partial \mathcal{L}}{\partial \lambda} = -(x + y - 1) = 0.$$

- From the first two equations, $2x + 2y = \lambda$, so $x + y = \frac{\lambda}{2}$.

- From the third equation, $x + y = 1$, so $\lambda = 2$.

- Substitute $\lambda = 2$ into the first equation:

$$2x + 2y = 2 \Rightarrow x + y = 1.$$

- Let $y = 1 - x$, and substitute into the objective function:

$$f(x, 1-x) = x^2 + (1-x)^2 + 2x(1-x) = x^2 + 1 - 2x + x^2 + 2x - 2x^2 = 1.$$

- The minimum value is 1, achieved when $x + y = 1$

4. Conclusion: the optimal solution is any (x, y) such that $x + y = 1$, with $f(x, y) = 1$.

Exercise 5: Convexity of the Log-Sum-Exp Function

Problem:

Show that the log-sum-exp function $f(x, y) = \log(e^x + e^y)$ is convex.

Solution:

1. Second-Order Condition:

Compute the Hessian of $f(x, y) = \log(e^x + e^y)$.

2. First Derivatives:

$$\frac{\partial f}{\partial x} = \frac{e^x}{e^x + e^y}, \quad \frac{\partial f}{\partial y} = \frac{e^y}{e^x + e^y}.$$

3. Second Derivatives:

$$\frac{\partial^2 f}{\partial x^2} = \frac{e^x(e^x + e^y) - e^{2x}}{(e^x + e^y)^2} = \frac{e^x e^y}{(e^x + e^y)^2},$$

$$\frac{\partial^2 f}{\partial y^2} = \frac{e^y e^x}{(e^x + e^y)^2},$$

$$\frac{\partial^2 f}{\partial x \partial y} = -\frac{e^x e^y}{(e^x + e^y)^2}.$$

4. Hessian Matrix:

$$\nabla^2 f(x, y) = \frac{e^x e^y}{(e^x + e^y)^2} \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix}.$$

5. Positive Semi-definiteness:

The eigenvalues of the matrix $\begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix}$ are 0 and 2, so the Hessian is positive semi-definite. The log-sum-exp function $f(x, y) = \log(e^x + e^y)$ is convex.

Chapter 4: Identification of Optimum of a Function

4.1 Introduction to Optimization and Optima

In this chapter, we focus on the identification of the optimum of a function, which is a fundamental concept in optimization. The goal is to find the minimum or maximum of a function, depending on the problem at hand. We will explore various methods for identifying optima, including analytical methods, numerical methods, and gradient-based approaches. The chapter will also cover the conditions for optimality, such as the first-order and second-order conditions, and discuss the challenges of finding global optima in the presence of multiple local optima.

4.2 Types of Optima

4.2.1 Local Optimum

A point x^* is called a local minimum of a function $f(x)$ if there exists a neighborhood around x^* such that: $f(x^*) \leq f(x), \quad \forall x$ in the neighborhood.

Similarly, x^* is a local maximum if:

$$f(x^*) \geq f(x), \quad \forall x \text{ in the neighborhood.}$$

4.2.2 Global Optimum

A point x^* is called a global minimum of a function $f(x)$ if:

$$f(x^*) \leq f(x), \quad \forall x \in \text{dom}(f).$$

Similarly, x^* is a global maximum if:

$$f(x^*) \geq f(x), \quad \forall x \in \text{dom}(f).$$

4.2.3 Saddle Points

A saddle point is a point where the function has a local minimum in one direction and a local maximum in another direction. Saddle points are common in high-dimensional optimization problems.

4.3 Conditions for Optimality

4.3.1 First-Order Necessary Condition

For a differentiable function $f(x)$, if x^* is a local minimum or maximum, then the gradient of f at x^* must be zero:

$$\nabla f(x^*) = 0.$$

This condition is necessary but not sufficient for optimality.

Example:

- For $f(x) = x^2$, the gradient is $f'(x) = 2x$

. Setting $f'(x) = 0$ gives $x = 0$, which is the global minimum.

4.3.2 Second-Order Sufficient Condition

For a twice-differentiable function $f(x)$, if x^* satisfies:

$$\nabla f(x^*) = 0 \text{ and } \nabla^2 f(x) \succ 0$$

Then x^* is a local minimum. Similarly, if $\nabla^2 f(x^*) \prec 0$, then x^* is a local maximum.

Example:

- For $f(x) = x^2$, the Hessian is $f''(x) = 2$, which is positive definite. Thus, $x = 0$

is a local minimum.

4.3.3 Global Optimality

For convex functions, any local minimum is also a global minimum. This property makes convex optimization problems particularly tractable.

Example:

- The function $f(x) = x^2$ is convex, so its local minimum at $x = 0$ is also the global minimum.

4.4 Analytical Methods for Finding Optima

4.4.1 Direct Solution of Gradient Equations

For simple functions, the gradient equations $\nabla f(x) = 0$ can be solved analytically to find critical points.

Example:

- For $f(x, y) = x^2 + y^2$, the gradient equations are:

$$\frac{\partial f}{\partial x} = 2x = 0, \quad \frac{\partial f}{\partial y} = 2y = 0.$$

The solution is $x = 0$, $y = 0$, which is the global minimum.

4.4.2 Lagrange Multipliers

The Lagrange multiplier method is used to find the optima of a function subject to equality constraints. The Lagrangian is defined as:

$$\mathcal{L}(x, \lambda) = f(x) + \lambda^T g(x),$$

where $g(x) = 0$ represents the constraints.

Example:

- Minimize $f(x, y) = x^2 + y^2$ subject to $x + y = 1$.

- The Lagrangian is:

$$\mathcal{L}(x, y, \lambda) = x^2 + y^2 + \lambda(x + y - 1).$$

- The optimality conditions are:

$$\frac{\partial \mathcal{L}}{\partial x} = 2x + \lambda = 0,$$

$$\frac{\partial \mathcal{L}}{\partial y} = 2y + \lambda = 0,$$

$$\frac{\partial \mathcal{L}}{\partial \lambda} = x + y - 1 = 0.$$

- Solving these equations gives $x = 0.5$, $y = 0.5$, and $\lambda = -1$.

4.5 Numerical Methods for Finding Optima

4.5.1 Gradient Descent

Gradient descent is an iterative optimization algorithm used to find the minimum of a function. The update rule is:

$$x_{k+1} = x_k - \alpha \nabla f(x_k),$$

where α is the learning rate.

Example:

- Minimize $f(x) = x^2$ using gradient descent.

- The gradient is $f'(x) = 2x$.

- Starting from $x_0 = 1$, the updates are:

$$x_{k+1} = x_k - \alpha \cdot 2x_k.$$

- For $\alpha = 0.1$, the sequence converges to $x = 0$.

4.5.2 Newton's Method

Newton's method is a second-order optimization algorithm that uses the Hessian matrix to find the minimum of a function. The update rule is:

$$x_{k+1} = x_k - [\nabla^2 f(x_k)]^{-1} \nabla f(x_k).$$

Example:

- Minimize $f(x) = x^2$ using Newton's method.

- The gradient is $f'(x) = 2x$, and the Hessian is $f''(x) = 2$.

- Starting from $x_0 = 1$, the updates are:

$$x_{k+1} = x_k - \frac{2x_k}{2} = 0.$$

- The method converges in one step.

4.5.3 Quasi-Newton Methods

Quasi-Newton methods, such as the BFGS algorithm, approximate the Hessian matrix to reduce computational cost while maintaining fast convergence.

Example:

- Minimize $f(x, y) = x^2 + y^2$ using the BFGS algorithm.

- The BFGS method approximates the Hessian and converges to the global minimum $(0, 0)$.

4.6 Challenges in Finding Global Optima

4.6.1 Multiple Local Optima

In non-convex optimization problems, there may be multiple local optima, making it difficult to find the global optimum. Techniques such as random restarts and simulated annealing are used to explore the solution space.

Example:

- The function $f(x) = x^4 - 4x^3 + 4x^2$ has local minima at $x = 0$ and $x = 2$.

4.6.2 High-Dimensional Problems

In high-dimensional spaces, the curse of dimensionality makes it challenging to find global optima. Techniques such as dimensionality reduction and stochastic optimization are used to address this issue.

Example:

- In machine learning, high-dimensional optimization problems are common, such as training deep neural networks.

4.7 Applications of Optima Identification

4.7.1 Machine Learning

In machine learning, optimization is used to minimize the loss function during model training. Techniques such as gradient descent and stochastic gradient descent are widely used.

Example:

- Training a linear regression model involves minimizing the mean squared error (MSE) loss function.

4.7.2 Engineering Design

In engineering, optimization is used to design systems and components that meet performance criteria while minimizing cost or weight.

Example:

- Optimizing the shape of an aircraft wing to minimize drag.

4.7.3 Economics

In economics, optimization is used to model consumer behavior, resource allocation, and market equilibrium.

Example:

- Maximizing utility subject to a budget constraint.

Conclusion

The identification of the optimum of a function is a central problem in optimization. This chapter has covered the conditions for optimality, analytical and numerical methods for finding optima, and the challenges of global optimization. In the next chapter, we will explore non-linear optimization without constraints, focusing on methods such as the Newton method and steepest ascent method.

Exercises for Chapter 4: Identification of Optimum of a Function

Exercise 1: First-Order Necessary Condition

Problem:

Find the critical points of the function $f(x, y) = x^2 + y^2 + 2xy$ using the first-order necessary condition for optimality.

Solution:

1. First-Order Necessary Condition:

A critical point satisfies $\nabla f(x, y) = 0$.

2. Compute the Gradient:

For $f(x, y) = x^2 + y^2 + 2xy$, the gradient is:

$$\nabla f(x, y) = \begin{pmatrix} 2x + 2y \\ 2y + 2x \end{pmatrix}.$$

3. Set the Gradient to Zero:

$$2x + 2y = 0,$$

$$2y + 2x = 0.$$

These equations are equivalent, so we have: $x + y = 0.$

4. Critical Points:

Any point (x, y) such that $x + y = 0$ is a critical point. For example, $(0,0)$, $(1,-1)$, and $(-1,1)$ are critical points.

Exercise 2: Second-Order Sufficient Condition

Problem:

Determine whether the critical point $(0,0)$ of the function $f(x, y) = x^2 + y^2 + 2xy$ is a local minimum, local maximum, or saddle point using the second-order sufficient condition.

Solution:

1. Second-Order Sufficient Condition:

A critical point (x, y) is a local minimum if $\nabla^2 f(x, y) \succ 0$, and a local maximum if $\nabla^2 f(x, y) \prec 0$. Otherwise, it is a saddle point.

2. Compute the Hessian:

For $f(x, y) = x^2 + y^2 + 2xy$, the Hessian is:

$$\nabla^2 f(x, y) = \begin{pmatrix} 2 & 2 \\ 2 & 2 \end{pmatrix}.$$

3. Evaluate the Hessian at $(0,0)$:

$$\nabla^2 f(0,0) = \begin{pmatrix} 2 & 2 \\ 2 & 2 \end{pmatrix}.$$

4. Check Positive Definiteness:

- Compute the eigenvalues of $\nabla^2 f(0,0)$:

$$\det \begin{pmatrix} 2-\lambda & 2 \\ 2 & 2-\lambda \end{pmatrix} = 0.$$

Solve:

$$(2-\lambda)^2 - 4 = 0,$$

$$\lambda^2 - 4\lambda = 0,$$

$$\lambda = 0, 4.$$

The eigenvalues are 0 and 4. Since one eigenvalue is zero, the Hessian is not positive definite.

5. Conclusion: the critical point $(0,0)$ is a saddle point.

Exercise 3: Lagrange Multipliers

Problem:

Find the minimum of the function $f(x,y) = x^2 + y^2$ subject to the constraint $x + y = 1$, using the Lagrange multiplier method.

Solution:

1. Lagrangian Function:

The Lagrangian is:

$$\mathcal{L}(x,y,\lambda) = x^2 + y^2 + \lambda(x + y - 1).$$

2. First-Order Conditions:

$$\frac{\partial \mathcal{L}}{\partial x} = 2x + \lambda = 0,$$

$$\frac{\partial \mathcal{L}}{\partial y} = 2y + \lambda = 0,$$

$$\frac{\partial \mathcal{L}}{\partial \lambda} = x + y - 1 = 0.$$

3. Solve the System of Equations:

- From the first two equations:

$$2x + \lambda = 0 \quad \text{and} \quad 2y + \lambda = 0.$$

This implies $x = y$.

- Substitute $x = y$ into the constraint:

$$x + x = 1 \Rightarrow x = 0.5, \quad y = 0.5.$$

- Substitute $x = 0.5$ into the first equation:

$$2(0.5) + \lambda = 0 \Rightarrow \lambda = -1.$$

4. Optimal Solution:

The minimum of $f(x, y)$ subject to $x + y = 1$ is achieved at $(0.5, 0.5)$, with $f(0.5, 0.5) = 0.5$.

Exercise 4: Gradient Descent

Problem:

Minimize the function $f(x) = x^2$ using gradient descent with an initial guess $x_0 = 1$ and a learning rate $\alpha = 0.1$. Perform 5 iterations.

Solution:

1. Gradient Descent Update Rule:

$$x_{k+1} = x_k - \alpha f'(x_k).$$

For $f(x) = x^2$, the gradient is $f'(x) = 2x$.

2. Perform 5 Iterations:

- *Iteration* 1:

$$x_1 = x_0 - \alpha f'(x_0) = 1 - 0.1 \cdot 2(1) = 1 - 0.2 = 0.8.$$

- *Iteration* 2:

$$x_2 = 0.8 - 0.1 \cdot 2(0.8) = 0.8 - 0.16 = 0.64.$$

- *Iteration* 3:

$$x_3 = 0.64 - 0.1 \cdot 2(0.64) = 0.64 - 0.128 = 0.512.$$

- *Iteration* 4:

$$x_4 = 0.512 - 0.1 \cdot 2(0.512) = 0.512 - 0.1024 = 0.4096.$$

- *Iteration* 5:

$$x_5 = 0.4096 - 0.1 \cdot 2(0.4096) = 0.4096 - 0.08192 = 0.32768.$$

3. Conclusion: after 5 iterations, the value of x is 0.32768, which is approaching the global minimum at $x = 0$.

Exercise 5: Newton's Method

Problem:

Minimize the function $f(x) = x^2$ using Newton's method with an initial guess $x_0 = 1$. Perform 2 iterations.

Solution:

1. Newton's Method Update Rule:

$$x_{k+1} = x_k - \frac{f'(x_k)}{f''(x_k)}.$$

For $f(x) = x^2$, the gradient is $f'(x) = 2x$, and the Hessian is $f''(x) = 2$.

2. Perform 2 Iterations:

- *Iteration* 1:

$$x_1 = x_0 - \frac{f'(x_0)}{f''(x_0)} = 1 - \frac{2(1)}{2} = 1 - 1 = 0.$$

- *Iteration* 2:

$$x_2 = x_1 - \frac{f'(x_1)}{f''(x_1)} = 0 - \frac{2(0)}{2} = 0.$$

3. Conclusion: Newton's method converges to the global minimum $x = 0$ in one iteration.

Exercise 6: Global Optima in Non-Convex Functions

Problem:

Consider the function $f(x) = x^4 - 4x^3 + 4x^2$. Find all critical points and determine whether they are local minima, local maxima, or saddle points.

Solution:

1. First-Order Necessary Condition:

The gradient is: $f'(x) = 4x^3 - 12x^2 + 8x.$

Set $f'(x) = 0$:

$$4x^3 - 12x^2 + 8x = 0 \Rightarrow x(x^2 - 3x + 2) = 0.$$

The critical points are $x = 0$, $x = 1$, and $x = 2$.

2. Second-Order Sufficient Condition:

The second derivative is:

$$f''(x) = 12x^2 - 24x + 8.$$

- *At* $x = 0$:

$$f''(0) = 8 > 0 \Rightarrow \text{local minimum.}$$

- *At* $x = 1$:

$$f''(1) = 12 - 24 + 8 = -4 < 0 \Rightarrow \text{local maximum.}$$

- *At* $x = 2$:

$$f''(2) = 48 - 48 + 8 = 8 > 0 \Rightarrow \text{local minimum.}$$

3. Conclusion:

- $x = 0$ is a local minimum.
- $x = 1$ is a local maximum.
- $x = 2$ is a local minimum.

Exercise7. For the following functions, find the critical point(s):

(a) $y = 2x^3 - 0.5x^2 + 2$,

(b) $y = 4x_1^2 - x_1x_2 + x_2^2 - x_1^3$,

(c) $y = 2x_1^2 + x_2^2 + 4x_3^2 - x_1 + 2x_3$.

Determine whether the critical points are local maximums, minimums, or saddle points.

Solution:

(a) $y = 2x^3 - 0.5x^2 + 2$,

First-order condition (FOC): $\frac{dy}{dx} = 6x^2 - x = 0.$

Critical values: $x = 0, \frac{1}{6}$.

Second-order condition (SOC): $\frac{d^2y}{dx^2} = 12x - 1.$

- At $x = 0$:

$$\left.\frac{d^2y}{dx^2}\right|_{x=0} = -1 < 0 \quad \text{(local maximum).}$$

- At $x = \frac{1}{6}$:

$$\left.\frac{d^2y}{dx^2}\right|_{x=1/6} = 1 > 0 \quad \text{(local minimum).}$$

(b) $y = 4x_1^2 - x_1x_2 + x_2^2 - x_1^3,$

FOCs: $\frac{\partial y}{\partial x_1} = 8x_1 - x_2 - 3x_1^2 = 0,$

$$\frac{\partial y}{\partial x_2} = -x_1 + 2x_2 = 0.$$

From the second FOC, $x_1 = 2x_2$. Substitute into the first FOC:

$$8(2x_2) - x_2 - 3(2x_2)^2 = 0 \Rightarrow 16x_2 - x_2 - 12x_2^2 = 0 \Rightarrow x_2(15 - 12x_2) = 0.$$

Critical points:

1. $(x_1, x_2) = (0, 0),$

2. $(x_1, x_2) = \left(\frac{2}{15}, \frac{1}{15}\right).$

Hessian matrix: $H = \begin{bmatrix} 8 - 6x_1 & -1 \\ -1 & 2 \end{bmatrix}.$

- At $(0,0)$: $H = \begin{bmatrix} 8 & -1 \\ -1 & 2 \end{bmatrix}.$

Principal minors: $8>0$ and $15>0 \Rightarrow$ local minimum.

- At $\left(\frac{2}{15},\frac{1}{15}\right)$: $H=\begin{bmatrix}\frac{36}{5} & -1\\ -1 & 2\end{bmatrix}$.

Principal minors: $\frac{36}{5}>0$ and $\frac{67}{5}>0 \Rightarrow$ local minimum.

Evaluating the function at these points:

- $y(0,0)=0$ (global minimum),

- $y\left(\frac{2}{15},\frac{1}{15}\right)=0.064$ (local minimum).

(c) $y=2x_1^2+x_2^2+4x_3^2-x_1+2x_3$,

FOCs: $\frac{\partial y}{\partial x_1}=4x_1-1=0\Rightarrow x_1=\frac{1}{4}$,

$\frac{\partial y}{\partial x_2}=2x_2=0\Rightarrow x_2=0$,

$\frac{\partial y}{\partial x_3}=8x_3+2=0\Rightarrow x_3=-\frac{1}{4}$.

Critical point: $(x_1,x_2,x_3)=\left(\frac{1}{4},0,-\frac{1}{4}\right)$.

Hessian matrix: $H=\begin{bmatrix}4 & 0 & 0\\ 0 & 2 & 0\\ 0 & 0 & 8\end{bmatrix}$.

Principal minors: $4>0$, $8>0$, and $64>0 \Rightarrow$ global minimum.

Exercise 8. A publisher pays the author of a book a royalty of 15%. Demand for the book is $x=200-5p$, and the production cost is $C=10+2x+x^2$. Find the optimal sales from both the author's and the publisher's perspective. Prove that the level of sales you find for each agent is a maximum.

Solution:

Author's perspective:

The author maximizes royalty revenue: $\max_x 0.15px$.

Inverse demand: $p = 40 - 0.2x$.

Objective function: $\max_x 0.15(40 - 0.2x)x = 6x - 0.03x^2$.

FOC: $6 - 0.06x = 0 \Rightarrow x = 100$.

SOC: $-0.06 < 0$ (maximum).

Publisher's perspective:

The publisher maximizes profit: $\max_x px - (10 + 2x + x^2) - 0.15px$.

Objective function:

$$\max_x 0.85(40 - 0.2x)x - (10 + 2x + x^2) = 34x - 0.17x^2 - 10 - 2x - x^2.$$

FOC: $34 - 0.34x - 2 - 2x = 0 \Rightarrow x = 13.67$.

SOC: $-2.34 < 0$ (maximum).

Exericse9. A monopoly supplies its markets from two plants, with cost functions: $C_1 = q_1^2$, $C_2 = 2q_2$. The monopoly faces a linear demand curve: $p = 70 - 2(q_1 + q_2)$. Find the firm's profit-maximizing output for each plant and the highest profit level that the monopolist can achieve. Does the profit-maximizing output level for each plant satisfy the second-order conditions for a maximum?

Solution:

The monopolist's profit function: $\max_{q_1, q_2} (70 - 2(q_1 + q_2))(q_1 + q_2) - q_1^2 - 2q_2$.

FOCs:

$$70 - 4(q_1 + q_2) - 2q_1 = 0,$$

$$70 - 4(q_1 + q_2) - 2 = 0.$$

Solving: $q_1 = 1$, $q_2 = 16$.

Total profit: 579.

Hessian matrix: $H = \begin{bmatrix} -6 & -4 \\ -4 & -4 \end{bmatrix}$.

Principal minors: $-6 < 0$ and $8 > 0 \Rightarrow$ maximum.

Exercise 10. A monopoly seller divides its market into two submarkets with inverse demand functions: $p_1 = 100 - q_1$, $p_2 = 120 - 2q_2$. The cost function is $C = 20(q_1 + q_2)$. Solve for the optimal output and price levels in each submarket. Prove that the monopolist is maximizing profit.

Solution:

The monopolist's profit function:

$$\max_{q_1, q_2} (100 - q_1)q_1 + (120 - 2q_2)q_2 - 20(q_1 + q_2).$$

FOCs:

$$100 - 2q_1 - 20 = 0 \Rightarrow q_1 = 40,$$

$$120 - 4q_2 - 20 = 0 \Rightarrow q_2 = 25.$$

Prices: $p_1 = 60$, $p_2 = 70$.

Hessian matrix: $H = \begin{bmatrix} -2 & 0 \\ 0 & -4 \end{bmatrix}$.

Principal minors: $-2 < 0$ and $8 > 0 \Rightarrow$ maximum.

Chapter 5: Non-Linear Optimization Without Constraints

5.1 Introduction to Non-Linear Optimization

In this chapter, we focus on non-linear optimization without constraints, which involves finding the minimum or maximum of a non-linear function without any restrictions on the variables. Non-linear optimization problems are more complex than linear ones because the objective function can have multiple local optima, and the solution methods often require iterative approaches. We will explore several key methods for solving such problems, including the Newton method, steepest descent method, and quasi-Newton methods.

5.2 Newton's Method

5.2.1 Overview of Newton's Method

Newton's method is a second-order optimization algorithm that uses the Hessian matrix (second derivative) to find the minimum or maximum of a function. It is particularly effective for smooth, well-behaved functions.

Key Idea:

Newton's method approximates the function locally by a quadratic function and finds the minimum of this quadratic approximation.

5.2.2 Algorithm

1. Initialization:

   Start with an initial guess x_0.

2. Iteration:

At each iteration k, update the solution using:

$$x_{k+1} = x_k - [\nabla 2 f(x_k)] - 1 \nabla f(x_k),$$

where:

- $\nabla f(x_k)$ is the gradient of f at x_k.

- $\nabla 2 f(x_k)$ is the Hessian matrix of f at x_k.

3. Stopping Criterion:

Stop when the gradient $\nabla f(x_k)$ is sufficiently small or when the change in x_k is below a tolerance level.

5.2.3 Advantages and Disadvantages

- *Advantages:*

- Fast convergence near the optimum (quadratic convergence).

- Works well for smooth, well-behaved functions.

- *Disadvantages:*

- Requires computation of the Hessian matrix, which can be expensive for high-dimensional problems.

- May converge to a saddle point or diverge if the initial guess is poor.

5.2.4 Example

Problem:

Minimize the function $f(x) = x4 - 4x3 + 4x2$ using Newton's method with an initial guess $x_0 = 1$.

Solution:

1. Gradient and Hessian:

$$\nabla f(x) = 4x3 - 12x2 + 8x,$$

$$\nabla 2 f(x) = 12x2 - 24x + 8.$$

2. Newton's Method Iteration:

- *Iteration 1:*

$$x_1 = x_0 - \frac{\nabla f(x_0)}{\nabla 2 f(x_0)} = 1 - \frac{4(1)3 - 12(1)2 + 8(1)}{12(1)2 - 24(1) + 8} = 1 - \frac{0}{-4} = 1.$$

- *Iteration 2:*

$$x_2 = x_1 - \frac{\nabla f(x_1)}{\nabla 2 f(x_1)} = 1 - \frac{0}{-4} = 1.$$

3. Conclusion:

Newton's method converges to $x = 1$, which is a local maximum.

5.3 Steepest Descent Method

5.3.1 Overview of Steepest Descent

The steepest descent method is a first-order optimization algorithm that iteratively moves in the direction of the negative gradient to find the minimum of a function.

Key Idea:

At each iteration, the algorithm takes a step in the direction of the steepest decrease of the function.

5.3.2 Algorithm

1. Initialization:

Start with an initial guess x_0.

2. Iteration:

At each iteration *k*, update the solution using:

$$x_{k+1} = x_k - \alpha_k \nabla f(x_k),$$

where:

- α_k is the step size (learning rate).

- $\nabla f(x_k)$ is the gradient of f at x_k.

3. Stopping Criterion:

Stop when the gradient $\nabla f(x_k)$ is sufficiently small or when the change in x_k is below a tolerance level.

5.3.3 Advantages and Disadvantages

- *Advantages:*

- Simple to implement.

- Does not require computation of the Hessian matrix.

- *Disadvantages:*

- Slow convergence (linear convergence).

- Sensitive to the choice of step size α_k.

5.3.4 Example

Problem:

Minimize the function $f(x) = x2$ using the steepest descent method with an initial guess $x_0 = 1$ and a fixed step size $\alpha = 0.1$. Perform 5 iterations.

Solution:

1. Gradient:

$\nabla f(x) = 2x.$

2. Steepest Descent Iteration:

- *Iteration 1:*

$x_1 = x_0 - \alpha \nabla f(x_0) = 1 - 0.1 \cdot 2(1) = 0.8.$

- *Iteration 2:*

$x_2 = 0.8 - 0.1 \cdot 2(0.8) = 0.64.$

- *Iteration 3:*

$x_3 = 0.64 - 0.1 \cdot 2(0.64) = 0.512.$

- *Iteration 4:*

$x_4 = 0.512 - 0.1 \cdot 2(0.512) = 0.4096.$

- *Iteration 5:*

$x_5 = 0.4096 - 0.1 \cdot 2(0.4096) = 0.32768.$

3. Conclusion:

After 5 iterations, the value of x is 0.32768 , which is approaching the global minimum at $x = 0$.

5.4 Quasi-Newton Methods

5.4.1 Overview of Quasi-Newton Methods

Quasi-Newton methods are a family of optimization algorithms that approximate the Hessian matrix to avoid the computational cost of computing it directly. The most popular quasi-Newton method is the BFGS algorithm.

Key Idea:

Quasi-Newton methods build an approximation of the Hessian matrix using gradient information from previous iterations.

5.4.2 BFGS Algorithm

1. Initialization:

Start with an initial guess x_0 and an initial Hessian approximation B_0 (often the identity matrix).

2. Iteration:

At each iteration *k:*

- Compute the search direction

$p_k = -B_k - 1\nabla f(x_k)$.

- Perform a line search to find the step size α_k .

- Update the solution:

$x_{k+1} = x_k + \alpha_k p_k$.

- Update the Hessian approximation B_{k+1} using the BFGS update formula.

3. Stopping Criterion:

Stop when the gradient $\nabla f(x_k)$ is sufficiently small or when the change in x_k is below a tolerance level.

5.4.3 Advantages and Disadvantages

- *Advantages:*

- Faster convergence than steepest descent.

- Does not require explicit computation of the Hessian matrix.

- *Disadvantages:*

- More complex to implement than steepest descent.

- Requires storage of the Hessian approximation, which can be large for high-dimensional problems.

5.4.4 Example

Problem:

Minimize the function $f(x, y) = x2 + y2 + 2xy$ using the BFGS algorithm with an initial guess $(x_0, y_0) = (1,1)$.

Solution:

1. Gradient:

$$\nabla f(x, y) = \begin{pmatrix} 2x + 2y \\ 2y + 2x \end{pmatrix}.$$

2. BFGS Iteration:

- The BFGS algorithm will iteratively update the solution and the Hessian approximation until convergence.

3. Conclusion:

The BFGS algorithm will converge to the global minimum at (0,0).

Conclusion: Non-linear optimization without constraints is a fundamental problem in optimization, with applications in machine learning, engineering, economics, and more. This chapter has covered key methods for solving such problems, including Newton's method, the steepest descent method, and quasi-Newton methods. Each method has its strengths and weaknesses, and the choice of method depends on the specific problem and computational resources available.

Exercises for Chapter 5: Non-Linear Optimization Without Constraints

Exercise 1: Newton's Method

Problem:

Minimize the function $f(x) = x4 - 4x3 + 4x2$ using Newton's method with an initial guess $x_0 = 1$. Perform 2 iterations.

Solution:

1. Gradient and Hessian:

$$\nabla f(x) = 4x3 - 12x2 + 8x,$$

$$\nabla 2 f(x) = 12x2 - 24x + 8.$$

2. Newton's Method Iteration:

- *Iteration 1:*

$$x_1 = x_0 - \frac{\nabla f(x_0)}{\nabla 2 f(x_0)} = 1 - \frac{4(1)3 - 12(1)2 + 8(1)}{12(1)2 - 24(1) + 8} = 1 - \frac{0}{-4} = 1.$$

- *Iteration 2:*

$$x_2 = x_1 - \frac{\nabla f(x_1)}{\nabla 2 f(x_1)} = 1 - \frac{0}{-4} = 1.$$

3. Conclusion:

Newton's method converges to $x = 1$, which is a local maximum.

Exercise 2: Steepest Descent Method

Problem:

Minimize the function $f(x) = x2$ using the steepest descent method with an initial guess $x_0 = 1$ and a fixed step size $\alpha = 0.1$. Perform 5 iterations.

Solution:

1. Gradient:

$\nabla f(x) = 2x.$

2. Steepest Descent Iteration:

- Iteration 1:

$$x_1 = x_0 - \alpha \nabla f(x_0) = 1 - 0.1 \cdot 2(1) = 0.8.$$

- Iteration 2:

$x_2 = 0.8 - 0.1 \cdot 2(0.8) = 0.64.$

- Iteration 3:

$x_3 = 0.64 - 0.1 \cdot 2(0.64) = 0.512.$

- Iteration 4:

$x_4 = 0.512 - 0.1 \cdot 2(0.512) = 0.4096.$

- *Iteration 5:*

$x_5 = 0.4096 - 0.1 \cdot 2(0.4096) = 0.32768.$

Conclusion: After 5 iterations, the value of x is 0.32768, which is approaching the global minimum at $x = 0$.

Exercise 3: BFGS Algorithm

Problem:

Minimize the function $f(x, y) = x2 + y2 + 2xy$ using the BFGS algorithm with an initial guess $(x_0, y_0) = (1,1)$.

Solution:

1. Gradient:

$$\nabla f(x, y) = \begin{pmatrix} 2x + 2y \\ 2y + 2x \end{pmatrix}.$$

2. BFGS Iteration:

- The BFGS algorithm will iteratively update the solution and the Hessian approximation until convergence.

Conclusion: the BFGS algorithm will converge to the global minimum at $(0,0)$.

Exercise 4: Newton's Method for a Quadratic Function

Problem:

Minimize the function $f(x, y) = x2 + y2 + 2xy$ using Newton's method with an initial guess $(x_0, y_0) = (1,1)$. Perform 2 iterations.

Solution:

1. Gradient and Hessian:

$$\nabla f(x,y) = \begin{pmatrix} 2x+2y \\ 2y+2x \end{pmatrix},$$

$$\nabla 2f(x,y) = \begin{pmatrix} 2 & 2 \\ 2 & 2 \end{pmatrix}.$$

2. Newton's Method Iteration:

- *Iteration 1:*

$$\nabla f(1,1) = \begin{pmatrix} 4 \\ 4 \end{pmatrix},$$

$$\nabla 2f(1,1) = \begin{pmatrix} 2 & 2 \\ 2 & 2 \end{pmatrix}.$$

The inverse of the Hessian is:

$$[\nabla 2f(1,1)] - 1 = \begin{pmatrix} 0.5 & -0.5 \\ -0.5 & 0.5 \end{pmatrix}.$$

Update the solution:

$$\begin{pmatrix} x_1 \\ y_1 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \end{pmatrix} - \begin{pmatrix} 0.5 & -0.5 \\ -0.5 & 0.5 \end{pmatrix} \begin{pmatrix} 4 \\ 4 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \end{pmatrix} - \begin{pmatrix} 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}.$$

- *Iteration 2:*

The solution remains $(1,1)$.

Conclusion: Newton's method converges to $(1,1)$, which is a saddle point.

Exercise 5: Steepest Descent for a Quadratic Function

Problem: minimize the function $f(x,y) = x2 + y2 + 2xy$ using the steepest descent method with an initial guess $(x_0, y_0) = (1,1)$ and a fixed step size $\alpha = 0.1$. Perform 5 iterations.

Solution:

1. Gradient:

$$\nabla f(x,y)=\begin{pmatrix}2x+2y\\2y+2x\end{pmatrix}.$$

2. Steepest Descent Iteration:

- Iteration 1:

$$\nabla f(1,1)=\begin{pmatrix}4\\4\end{pmatrix},$$

$$\begin{pmatrix}x_1\\y_1\end{pmatrix}=\begin{pmatrix}1\\1\end{pmatrix}-0.1\begin{pmatrix}4\\4\end{pmatrix}=\begin{pmatrix}0.6\\0.6\end{pmatrix}.$$

- Iteration 2:

$$\nabla f(0.6,0.6)=\begin{pmatrix}2.4\\2.4\end{pmatrix},$$

$$\begin{pmatrix}x_2\\y_2\end{pmatrix}=\begin{pmatrix}0.6\\0.6\end{pmatrix}-0.1\begin{pmatrix}2.4\\2.4\end{pmatrix}=\begin{pmatrix}0.36\\0.36\end{pmatrix}.$$

- Iteration 3:

$$\nabla f(0.36,0.36)=\begin{pmatrix}1.44\\1.44\end{pmatrix},$$

$$\begin{pmatrix}x_3\\y_3\end{pmatrix}=\begin{pmatrix}0.36\\0.36\end{pmatrix}-0.1\begin{pmatrix}1.44\\1.44\end{pmatrix}=\begin{pmatrix}0.216\\0.216\end{pmatrix}.$$

- Iteration 4:

$$\nabla f(0.216,0.216)=\begin{pmatrix}0.864\\0.864\end{pmatrix},$$

$$\begin{pmatrix}x_4\\y_4\end{pmatrix}=\begin{pmatrix}0.216\\0.216\end{pmatrix}-0.1\begin{pmatrix}0.864\\0.864\end{pmatrix}=\begin{pmatrix}0.1296\\0.1296\end{pmatrix}.$$

- Iteration 5:

$$\nabla f(0.1296,0.1296)=\begin{pmatrix}0.5184\\0.5184\end{pmatrix},$$

$$\begin{pmatrix} x_5 \\ y_5 \end{pmatrix} = \begin{pmatrix} 0.1296 \\ 0.1296 \end{pmatrix} - 0.1 \begin{pmatrix} 0.5184 \\ 0.5184 \end{pmatrix} = \begin{pmatrix} 0.07776 \\ 0.07776 \end{pmatrix}.$$

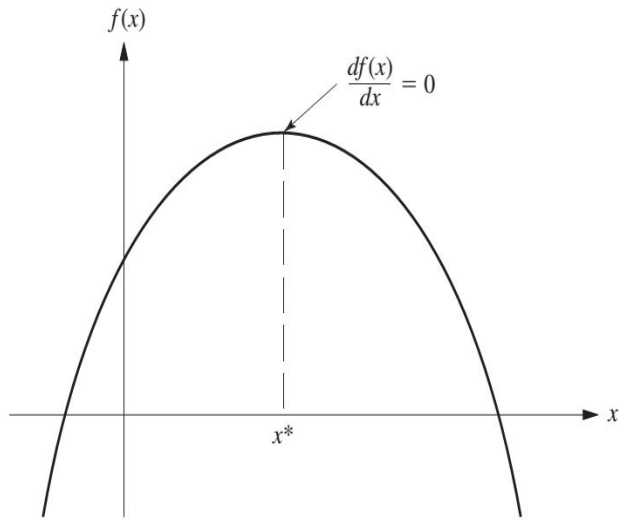
Conclusion: After 5 iterations, the solution is approaching the global minimum at $(0,0)$.

Newton & Bisection (continued)

We now begin discussing how to solve some of the types of problems just described by considering the simplest case—unconstrained optimization with just a single variable x ($n = 1$), where the differentiable function $f(x)$ to be maximized is concave. Thus, the necessary and sufficient condition for a particular solution $x = x^*$ to be optimal (a global maximum) is

$$\frac{df}{dx} = 0 \quad \text{at } x = x^*,$$

as depicted in figure below. If this equation can be solved directly for x^*, you are done. However, if $f(x)$ is not a particularly simple function, so the derivative is not just a linear or quadratic function, you may not be able to solve the equation analytically. If not, a number of search procedures are available for solving the problem numerically.



The approach with any of these search procedures is to find a sequence of trial solutions that leads toward an optimal solution. At each iteration, you begin at the current trial solution to conduct a systematic search that culminates by identifying a new improved trial solution. The procedure is continued until the trial solutions have converged to an optimal solution, assuming that one exists.

We now will describe two common search procedures. The first one (the bisection method) was chosen because it is such an intuitive and straightforward procedure. The second one (Newton's method) is included because it plays a fundamental role in nonlinear programming in general.

The Bisection Method

This search procedure always can be applied when $f(x)$ is concave (so that the second derivative is negative or zero for all x as depicted in Fig. 12.12. It also can be used for certain other functions as well. In particular, if x^* denotes the optimal solution, all that is needed is that

$$\frac{df(x)}{dx} > 0 \quad \text{if } x < x^*, \quad \frac{df(x)}{dx} = 0 \quad \text{if } x = x^*, \quad \frac{df(x)}{dx} < 0 \quad \text{if } x > x^*$$

These conditions automatically hold when $f(x)$ is concave, but they also can hold when the second derivative is positive for some (but not all) values of x.

The idea behind the bisection method is a very intuitive one, namely, that whether the slope (derivative) is positive or negative at a trial solution definitely indicates whether improvement lies immediately to the right or left, respectively. Thus, if the derivative evaluated at a particular value of x is positive, then x^* must be larger than this x (see Figure), so this x becomes a lower bound on the trial solutions that need to be considered thereafter. Conversely, if the derivative is negative, then x^* must be smaller than this x. so x would become an upper bound. Therefore, after both types of bounds have been identified, each new trial solution selected between the current bounds provides a new tighter bound of one type, thereby narrowing the search further. As long as a reasonable rule is used to select each trial solution in this way, the resulting sequence of trial solutions must converge to x^*. In practice, this means continuing the sequence until the distance between the bounds is sufficiently small that the next trial solution must be within a prespecified error tolerance of x^*.

This entire process is summarized next, given the notation

x' = current trial solution

$\overline{x}$ = current lower bound on x

$\overline{x}$ = current upper bound on x

ϵ = error tolerance for x

Although there are several reasonable rules for selecting each new trial solution, the one used in the bisection method is the midpoint rule (traditionally called the Bolzano search plan), which says simply to select the midpoint between the two current bounds.

Summary of the Bisection Method

Initialization: Select ϵ

. Find an initial $\overline{x}$ and $\underline{x}$

by inspection (or by respectively finding any value of x at which the derivative is positive and then negative). Select an initial trial solution

$$x' = \frac{\overline{x} + \underline{x}}{2}.$$

Iteration:

1. Evaluate $\frac{df(x)}{dx}$ at $x = x'$.

2. If $\frac{df(x)}{dx} \geq 0$, reset $\underline{x} = x'$.

3. If $\frac{df(x)}{dx} \leq 0$, reset $\overline{x} = x'$.

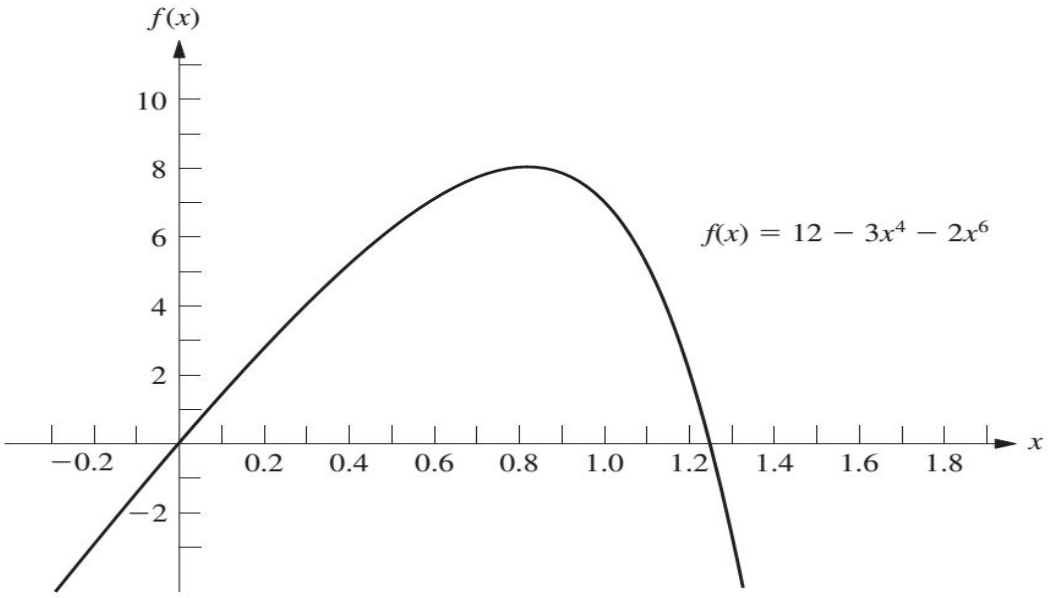
4. Select a new $x' = \frac{\overline{x} + \underline{x}}{2}$.

Stopping rule: If $\overline{x} - \underline{x} \leq 2\epsilon$, so that the new x' must be within ϵ of x^*, stop. Otherwise, perform another iteration.

Example of the Bisection Method

Suppose that the function to be maximized is $f(x) = 12x - 3x^4 - 2x^6$, as plotted in Figure below. Its first two derivatives are

$$\frac{df(x)}{dx} = 12(1 - x^3 - x^5), \quad \frac{d^2 f(x)}{dx^2} = -12(3x^2 + 5x^4).$$



Because the second derivative is nonpositive everywhere, $f(x)$ is a concave function, so the bisection method can be safely applied to find its global maximum (assuming a global maximum exists).A quick inspection of this function (without even constructing its graph as shown in Figure) indicates that $f(x)$ is positive for small positive values of x .but it is negative for $x < 0$ or $x > 2$. Therefore, $x = 0$ and $\overline{x} = 2$ can be used as the initial bounds, with their midpoint, $x' = 1$, as the initial trial solution. Let $\epsilon = 0.01$ be the error tolerance for x^* in the stopping rule, so the final $(\overline{x} - \underline{x}) \leq 0.02$ with the final x' at the midpoint.

Application of the bisection method to the example

Iteration	$\frac{df(x)}{dx}$	$\underline{x}$	$\bar{x}$	New x'	$f(x')$
0		0	2	1	7.0000
1	−12	0	1	0.5	5.7812
2	+10.12	0.5	1	0.75	7.6948
3	+4.09	0.75	1	0.875	7.8439
4	−2.19	0.75	0.875	0.8125	7.8672
5	+1.31	0.8125	0.875	0.84375	7.8829
6	−0.34	0.8125	0.84375	0.828125	7.8815
7	+0.51	0.828125	0.84375	0.8359375	7.8839
Stop					

Applying the bisection method then yields the sequence of results shown in Table 12.1. The conclusion is that

$$x \approx 0.836, \quad 0.828125 < x < 0.84375.$$

Newton's Method

Although the bisection method is an intuitive and straightforward procedure, it has the disadvantage of converging relatively slowly toward an optimal solution. Each iteration only decreases the difference between the bounds by one-half. Therefore, even with the fairly simple function being considered in Table 12.1, seven iterations were required to reduce the error tolerance for x^* to less than 0.01. Another seven iterations would be needed to reduce this error tolerance to less than 0.0001.

The basic reason for this slow convergence is that the only information about $f(x)$ being used is the value of the first derivative $f'(x)$ at the respective trial values of x. Additional helpful information can be obtained by considering the second derivative $f''(x)$ as well. This is what Newton's method does.

Summary of Newton's Method

Initialization: Select ϵ.

Find an initial trial solution x_i by inspection.Set $i = 1$.

Iteration i :

1. Calculate $f'(x_i)$, and $f''(x_i)$. Calculating $f(x_i)$ is optional.

2. Set $x_{i+1} = x_i - \dfrac{f'(x_i)}{f''(x_i)}$.

3. Stopping Rule: If $|x_{i+1} - x_i| \leq \epsilon$, stop; x_{i+1} is essentially the optimal solution. Otherwise, reset $i = i + 1$ and perform another iteration.

Example of Newton's Method

We now will apply Newton's method to the same example used for the bisection method. As depicted in Fig. 12.13, the function to be maximized is

$$f(x) = 12x - 3x^4 - 2x^6.$$

Thus, the formula for calculating the new trial solution (x_{i+1}) from the current one (x_i) is

$$x_{i+1} = x_i - \frac{f'(x_i)}{f''(x_i)} = x_i - \frac{12(1 - x^3 - x^5)}{-12(3x^2 + 5x^4)} = x_i + \frac{1 - x^3 - x^5}{3x^2 + 5x^4}.$$

After selecting $\epsilon = 0.00001$ and choosing $x_1 = 1$ as the initial trial solution, Table 12.2 shows the results from applying Newton's method to this example. After just four iterations, this method has converged to $x = 0.83762$ as the optimal solution with a very high degree of precision.

Application of Newton's method to the example

Iteration i	x_i	$f(x_i)$	$f'(x_i)$	$f''(x_i)$	x_{i+1}
1	1	7	−12	−96	0.875
2	0.875	7.8439	−2.1940	−62.733	0.84003
3	0.84003	7.8838	−0.1325	−55.279	0.83763
4	0.83763	7.8839	−0.0006	−54.790	0.83762

A comparison of this table with Table above illustrates how much more rapidly Newton's method converges than the bisection method. Nearly 20 iterations would be required for the bisection method to converge with the same degree of precision that Newton's method achieved after only four iterations.

Chapter 6: Steepest Ascent/Descent Method

6.1 Introduction to Steepest Ascent/Descent

The steepest ascent/descent method is a first-order optimization algorithm used to find the maximum (ascent) or minimum (descent) of a function. It is one of the simplest and most intuitive optimization techniques, relying on the gradient of the function to determine the direction of the steepest increase or decrease. In this chapter, we will explore the theoretical

foundations, algorithmic implementation, and practical applications of the steepest ascent/descent method.

6.2 Theoretical Foundations

6.2.1 Gradient and Direction of Steepest Ascent/Descent

The gradient of a function $f(x)$, denoted $\nabla f(x)$, is a vector that points in the direction of the steepest ascent of the function. Conversely, the negative gradient $-\nabla f(x)$ points in the direction of the steepest descent.

Mathematical Definition:

For a function $f:\mathbb{R}^n \to \mathbb{R}$, the gradient is:

$$\nabla f(x) = \begin{pmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \\ \vdots \\ \frac{\partial f}{\partial x_n} \end{pmatrix}.$$

Key Idea:

- Steepest Ascent: Move in the direction of $\nabla f(x)$ to maximize $f(x)$.

- Steepest Descent: Move in the direction of $-\nabla f(x)$ to minimize $f(x)$.

6.2.2 Step Size and Learning Rate

The step size (or learning rate), denoted α, determines how far to move in the direction of the gradient. A small step size may lead to slow convergence, while a large step size may cause oscillations or divergence.

Mathematical Update Rule:

For steepest descent, the update rule is: $x_{k+1} = x_k - \alpha \nabla f(x_k)$.

For steepest ascent, the update rule is: $x_{k+1} = x_k + \alpha \nabla f(x_k)$.

6.2.3 Convergence Properties

The steepest ascent/descent method is guaranteed to converge to a local optimum under certain conditions:

1. The function $f(x)$ is continuously differentiable.

2. The step size α is chosen appropriately (e.g., using line search).

However, the method may converge slowly, especially for functions with ill-conditioned Hessians (e.g., long, narrow valleys).

6.3 Algorithmic Implementation

6.3.1 Steepest Descent Algorithm

The steepest descent algorithm can be summarized as follows:

1. Initialization:

- Start with an initial guess x_0.

- Choose a step size α (or use line search to determine α dynamically).

2. Iteration:

- Compute the gradient $\nabla f(x_k)$.

- Update the solution: $x_{k+1} = x_k - \alpha \nabla f(x_k)$.

- Check for convergence (e.g., $||\nabla f(x_k)|| < \epsilon$).

3. Termination:

- Stop when the gradient is sufficiently small or when a maximum number of iterations is reached.

6.3.2 Line Search

Line search is a technique used to determine the optimal step size α at each iteration. The goal is to find α that minimizes (or maximizes) the function along the direction of the gradient.

Exact Line Search:

Find α that minimizes $f(x_k - \alpha \nabla f(x_k))$.

Inexact Line Search:

Use methods like the Armijo rule or Wolfe conditions to find a step size that provides sufficient decrease in the function value.

6.3.3 Variants of Steepest Descent

1. Fixed Step Size:

Use a constant step size α for all iterations.

2. Adaptive Step Size:

Adjust α dynamically based on the progress of the optimization.

3. Momentum-Based Methods:

Incorporate momentum to accelerate convergence (e.g., Nesterov accelerated gradient).

6.4 Practical Considerations

6.4.1 Choice of Step Size

The choice of step size α is critical for the performance of the steepest ascent/descent method:

- A small step size may lead to slow convergence.

- A large step size may cause oscillations or divergence.

Example:

For $f(x) = x^2$, a step size $\alpha = 0.1$ works well, but $\alpha = 1$ causes divergence.

6.4.2 Conditioned Problems

For functions with ill-conditioned Hessians, the steepest ascent/descent method may converge very slowly. In such cases, preconditioning or second-order methods (e.g., Newton's method) are more effective.

Example:

The function $f(x,y) = x^2 + 100y^2$ has an ill-conditioned Hessian, causing slow convergence for steepest descent.

6.4.3 Convergence Criteria

Common convergence criteria for the steepest ascent/descent method include:

1. Gradient Norm:

Stop when $\| \nabla f(x_k) \| < \epsilon$.

2. Function Value Change:

Stop when $| f(x_{k+1}) - f(x_k) | < \epsilon$.

3. Maximum Iterations:

Stop after a fixed number of iterations.

6.5 Applications of Steepest Ascent/Descent

6.5.1 Machine Learning

In machine learning, the steepest descent method is used to minimize the loss function during model training. Variants like stochastic gradient descent (SGD) are widely used for large-scale problems.

Example: Training a linear regression model by minimizing the mean squared error (MSE) loss function.

6.5.2 Signal Processing

In signal processing, the steepest ascent/descent method is used for tasks like filter design and adaptive filtering.

Example: Designing a filter to minimize the error between the desired and actual output.

6.6 Example Problems and Solutions

6.6.1 Example 1: Steepest Descent for a Quadratic Function

Problem:

Minimize the function $f(x,y)=x^2+y^2$ using the steepest descent method with an initial guess $(x_0,y_0)=(1,1)$ and a fixed step size $\alpha=0.1$. Perform 5 iterations.

Solution:

1. Gradient:

$$\nabla f(x,y)=\begin{pmatrix}2x\\2y\end{pmatrix}.$$

2. Steepest Descent Iteration:

- *Iteration 1:*

$$\nabla f(1,1)=\begin{pmatrix}2\\2\end{pmatrix},$$

$$\begin{pmatrix}x_1\\y_1\end{pmatrix}=\begin{pmatrix}1\\1\end{pmatrix}-0.1\begin{pmatrix}2\\2\end{pmatrix}=\begin{pmatrix}0.8\\0.8\end{pmatrix}.$$

- *Iteration 2:*

$$\nabla f(0.8,0.8)=\begin{pmatrix}1.6\\1.6\end{pmatrix},$$

$$\begin{pmatrix}x_2\\y_2\end{pmatrix}=\begin{pmatrix}0.8\\0.8\end{pmatrix}-0.1\begin{pmatrix}1.6\\1.6\end{pmatrix}=\begin{pmatrix}0.64\\0.64\end{pmatrix}.$$

- *Iteration 3:*

$$\nabla f(0.64,0.64)=\begin{pmatrix}1.28\\1.28\end{pmatrix},$$

$$\begin{pmatrix}x_3\\y_3\end{pmatrix}=\begin{pmatrix}0.64\\0.64\end{pmatrix}-0.1\begin{pmatrix}1.28\\1.28\end{pmatrix}=\begin{pmatrix}0.512\\0.512\end{pmatrix}.$$

- *Iteration 4:*

$$\nabla f(0.512,0.512)=\begin{pmatrix}1.024\\1.024\end{pmatrix},$$

$$\begin{pmatrix} x_4 \\ y_4 \end{pmatrix} = \begin{pmatrix} 0.512 \\ 0.512 \end{pmatrix} - 0.1 \begin{pmatrix} 1.024 \\ 1.024 \end{pmatrix} = \begin{pmatrix} 0.4096 \\ 0.4096 \end{pmatrix}.$$

- *Iteration 5:*

$$\nabla f(0.4096, 0.4096) = \begin{pmatrix} 0.8192 \\ 0.8192 \end{pmatrix},$$

$$\begin{pmatrix} x_5 \\ y_5 \end{pmatrix} = \begin{pmatrix} 0.4096 \\ 0.4096 \end{pmatrix} - 0.1 \begin{pmatrix} 0.8192 \\ 0.8192 \end{pmatrix} = \begin{pmatrix} 0.32768 \\ 0.32768 \end{pmatrix}.$$

Conclusion: after 5 iterations, the solution is approaching the global minimum at $(0,0)$.

6.6.2 *Example* 2: Steepest Ascent for a Non-Quadratic Function

Problem:

Maximize the function $f(x) = -x^4 + 4x^3 - 4x^2$ using the steepest ascent method with an initial guess $x_0 = 1$ and a fixed step size $\alpha = 0.1$. Perform 5 iterations.

Solution:

1. Gradient:

$$\nabla f(x) = -4x3 + 12x2 - 8x.$$

2. Steepest Ascent Iteration:

- *Iteration 1:*

$$x_1 = x_0 + \alpha \nabla f(x_0) = 1 + 0.1 \cdot (-4(1)3 + 12(1)2 - 8(1)) = 1 + 0.1 \cdot 0 = 1.$$

- *Iteration 2:*

$$x_2 = x_1 + \alpha \nabla f(x_1) = 1 + 0.1 \cdot 0 = 1.$$

- *Iteration 3:*

$$x_3 = x_2 + \alpha \nabla f(x_2) = 1 + 0.1 \cdot 0 = 1.$$

- *Iteration 4:*

$$x_4 = x_3 + \alpha \nabla f(x_3) = 1 + 0.1 \cdot 0 = 1.$$

- *Iteration 5:*

$$x_5 = x_4 + \alpha \nabla f(x_4) = 1 + 0.1 \cdot 0 = 1.$$

The steepest ascent method converges to $x = 1$, which is a local maximum.

Conclusion: *the steepest ascent/descent method is a fundamental optimization technique that relies on the gradient of the function to determine the direction of the steepest increase or decrease. While simple and intuitive, the method has limitations, such as slow convergence for ill-conditioned problems. In the next chapter, we will explore second-order optimization methods, such as Newton's method, which address some of these limitations.*

Exercise 3: Steepest Descent for a High-Dimensional Function

Problem:

Minimize the function $f(x, y, z) = x^2 + y^2 + z^2$ using the steepest descent method with an initial guess $(x_0, y_0, z_0) = (1,1,1)$ and a fixed step size $\alpha = 0.1$. Perform 5 iterations.

Solution:

1. Gradient:

$$\nabla f(x, y, z) = \begin{pmatrix} 2x \\ 2y \\ 2z \end{pmatrix}.$$

2. Steepest Descent Iteration:

- *Iteration 1:*

$$\nabla f(1,1,1) = \begin{pmatrix} 2 \\ 2 \\ 2 \end{pmatrix},$$

$$\begin{pmatrix} x_1 \\ y_1 \\ z_1 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} - 0.1 \begin{pmatrix} 2 \\ 2 \\ 2 \end{pmatrix} = \begin{pmatrix} 0.8 \\ 0.8 \\ 0.8 \end{pmatrix}.$$

- *Iteration 2:*

$$\nabla f(0.8, 0.8, 0.8) = \begin{pmatrix} 1.6 \\ 1.6 \\ 1.6 \end{pmatrix},$$

$$\begin{pmatrix} x_2 \\ y_2 \\ z_2 \end{pmatrix} = \begin{pmatrix} 0.8 \\ 0.8 \\ 0.8 \end{pmatrix} - 0.1 \begin{pmatrix} 1.6 \\ 1.6 \\ 1.6 \end{pmatrix} = \begin{pmatrix} 0.64 \\ 0.64 \\ 0.64 \end{pmatrix}.$$

- *Iteration 3:*

$$\nabla f(0.64, 0.64, 0.64) = \begin{pmatrix} 1.28 \\ 1.28 \\ 1.28 \end{pmatrix},$$

$$\begin{pmatrix} x_3 \\ y_3 \\ z_3 \end{pmatrix} = \begin{pmatrix} 0.64 \\ 0.64 \\ 0.64 \end{pmatrix} - 0.1 \begin{pmatrix} 1.28 \\ 1.28 \\ 1.28 \end{pmatrix} = \begin{pmatrix} 0.512 \\ 0.512 \\ 0.512 \end{pmatrix}.$$

- *Iteration 4:*

$$\nabla f(0.512, 0.512, 0.512) = \begin{pmatrix} 1.024 \\ 1.024 \\ 1.024 \end{pmatrix},$$

$$\begin{pmatrix} x_4 \\ y_4 \\ z_4 \end{pmatrix} = \begin{pmatrix} 0.512 \\ 0.512 \\ 0.512 \end{pmatrix} - 0.1 \begin{pmatrix} 1.024 \\ 1.024 \\ 1.024 \end{pmatrix} = \begin{pmatrix} 0.4096 \\ 0.4096 \\ 0.4096 \end{pmatrix}.$$

- *Iteration 5:*

$$\nabla f(0.4096, 0.4096, 0.4096) = \begin{pmatrix} 0.8192 \\ 0.8192 \\ 0.8192 \end{pmatrix},$$

$$\begin{pmatrix} x_5 \\ y_5 \\ z_5 \end{pmatrix} = \begin{pmatrix} 0.4096 \\ 0.4096 \\ 0.4096 \end{pmatrix} - 0.1 \begin{pmatrix} 0.8192 \\ 0.8192 \\ 0.8192 \end{pmatrix} = \begin{pmatrix} 0.32768 \\ 0.32768 \\ 0.32768 \end{pmatrix}.$$

Conclusion: after 5 iterations, the solution is approaching the global minimum at $(0,0,0)$.

Chapter 7: The Jacobian Method in Nonlinear Programming

7.1 Introduction to the Jacobian Method

The Jacobian method is a powerful tool in nonlinear programming, particularly useful for solving systems of nonlinear equations and optimizing functions with multiple variables. Named after the mathematician Carl Gustav Jacob Jacobi, the Jacobian matrix plays a central role in this method. It is a matrix of all first-order partial derivatives of a vector-valued function, providing crucial information about the local behavior of the function.

In this chapter, we will explore the theoretical foundations of the Jacobian method, its applications in nonlinear programming, and its relevance to quantitative economics. We will also discuss practical examples and computational techniques to help you understand and apply this method effectively.

7.2 The Jacobian Matrix

7.2.1 Definition and Construction

Consider a vector-valued function $\mathbf{F}: \mathbb{R}^n \to \mathbb{R}^m$, where $\mathbf{F}(\mathbf{x}) = [f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_m(\mathbf{x})]T$. The Jacobian matrix $\mathbf{J}(\mathbf{x})$ of $\mathbf{F}$ is an $m \times n$ matrix defined as:

$$\mathbf{J}(\mathbf{x}) = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \frac{\partial f_m}{\partial x_2} & \cdots & \frac{\partial f_m}{\partial x_n} \end{bmatrix}$$

Each element $\frac{\partial f_i}{\partial x_j}$ represents the partial derivative of the *i-th* function with respect to the *j-th* variable.

7.2.2 Interpretation

The Jacobian matrix provides a linear approximation of the function $\mathbf{F}$ near a point $\mathbf{x}$. It captures the rate of change of each component of $\mathbf{F}$ with respect to each variable, offering insights into the local behavior of the function.

7.3 Applications in Nonlinear Programming

7.3.1 Solving Systems of Nonlinear Equations

One of the primary applications of the Jacobian method is in solving systems of nonlinear equations. Given a system of *m* nonlinear equations in *n* variables: $\mathbf{F}(\mathbf{x}) = \mathbf{0}$. We can use the Jacobian matrix to apply iterative methods such as Newton's method. The iterative update rule is given by:

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \mathbf{J}^{-1}(\mathbf{x}_k)\mathbf{F}(\mathbf{x}_k)$$

Where $\mathbf{J}^{-1}(\mathbf{x}_k)$ is the inverse of the Jacobian matrix at $\mathbf{x}_k$.

7.3.2 Optimization Problems

In optimization, the Jacobian method is used to find the critical points of a function. For a scalar-valued function $f : \mathbb{R}^n \to \mathbb{R}$, the gradient ∇f is a special case of the Jacobian matrix. The critical points are found by solving:

$$\nabla f(\mathbf{x}) = \mathbf{0}$$

For constrained optimization problems, the Jacobian matrix of the constraint functions is used to form the Lagrangian and derive the Karush-Kuhn-Tucker (KKT) conditions.

7.4 Practical Examples in Quantitative Economics

7.4.1 Utility Maximization

Consider a consumer maximizing utility $U(x_1, x_2)$ subject to a budget constraint

$$p_1 x_1 + p_2 x_2 = m$$

. The Lagrangian is:

$$\mathcal{L}(x_1, x_2, \lambda) = U(x_1, x_2) - \lambda(p_1 x_1 + p_2 x_2 - m)$$

The first-order conditions are:

$$\frac{\partial \mathcal{L}}{\partial x_1} = \frac{\partial U}{\partial x_1} - \lambda p_1 = 0 \frac{\partial \mathcal{L}}{\partial x_2} = \frac{\partial U}{\partial x_2} - \lambda p_2 = 0 \frac{\partial \mathcal{L}}{\partial \lambda} = -(p_1 x_1 + p_2 x_2 - m) = 0$$

The Jacobian matrix of this system can be used to solve for the optimal consumption bundle (x_1, x_2) and the Lagrange multiplier λ.

7.4.2 Production Function Optimization

A firm aims to maximize output $Q(K, L)$

given input prices *r* and *w*. The optimization problem is:

$$\max_{K,L} Q(K, L) \quad \text{subject to} \quad rK + wL = C$$

The Lagrangian and first-order conditions are:

$$\mathcal{L}(K, L, \lambda) = Q(K, L) - \lambda(rK + wL - C)$$

$$\frac{\partial \mathcal{L}}{\partial K} = \frac{\partial Q}{\partial K} - \lambda r = 0 \frac{\partial \mathcal{L}}{\partial L} = \frac{\partial Q}{\partial L} - \lambda w = 0 \frac{\partial \mathcal{L}}{\partial \lambda} = -(rK + wL - C) = 0$$

The Jacobian matrix of this system helps in finding the optimal input levels (K, L) and the shadow price λ.

7.5 Computational Techniques

7.5.1 Numerical Differentiation

When analytical derivatives are difficult to compute, numerical differentiation techniques can be used to approximate the Jacobian matrix. Common methods include finite differences and automatic differentiation.

7.5.2 Software Implementation

Modern software tools such as MATLAB, Python (with libraries like NumPy and SciPy), and R provide built-in functions to compute the Jacobian matrix and solve nonlinear systems. These tools are essential for practical applications in quantitative economics.

Conclusion: *the Jacobian method is a cornerstone of nonlinear programming, offering a robust framework for solving complex optimization problems and systems of nonlinear equations. Its applications in quantitative economics are vast, ranging from utility maximization to production optimization. By mastering the Jacobian method, you will be well-equipped to tackle a wide array of problems in your field.*

7.7 Exercises

1. Compute the Jacobian matrix for the function $\mathbf{F}(x, y) = [x^2 + y^2, xy]^t$.

2. Use the Jacobian method to solve the system of equations:

$$x^2 + y^2 = 1 \quad , \quad x + y = 1$$

3. Apply the Jacobian method to find the critical points of the function

$$f(x, y) = x^3 + y^3 - 3xy\,.$$

7.9 Additional Examples

7.9.1 Example: Nonlinear System of Equations

Consider the following system of nonlinear equations:

$$f_1(x, y) = x^2 + y^2 - 4 = 0 \quad , \; f_2(x, y) = x^2 - y - 1 = 0$$

The Jacobian matrix for this system is:

$$\mathbf{J}(x,y) = \begin{bmatrix} \frac{\partial f_1}{\partial x} & \frac{\partial f_1}{\partial y} \\ \frac{\partial f_2}{\partial x} & \frac{\partial f_2}{\partial y} \end{bmatrix} = \begin{bmatrix} 2x & 2y \\ 2x & -1 \end{bmatrix}$$

Using the Jacobian method (Newton's method), we can iteratively solve for (x, y).

7.9.2 Example: Constrained Optimization

Consider the problem of maximizing a Cobb-Douglas production function:

$$Q(K,L) = K^{0.5}L^{0.5}$$

subject to the budget constraint: $2K + 3L = 100$

The Lagrangian is:

$$\mathcal{L}(K,L,\lambda) = K^{0.5}L^{0.5} - \lambda(2K + 3L - 100)$$

The first-order conditions are:

$$\frac{\partial \mathcal{L}}{\partial K} = 0.5K^{-0.5}L^{0.5} - 2\lambda = 0 \quad , \frac{\partial \mathcal{L}}{\partial L} = 0.5K^{0.5}L^{-0.5} - 3\lambda = 0 \, , \frac{\partial \mathcal{L}}{\partial \lambda} = -(2K + 3L - 100) = 0$$

The Jacobian matrix of this system can be used to solve for K, L, and λ.

7.11 Implementation in LINGO

LINGO is a powerful tool for solving optimization problems, including nonlinear programming problems. Below are examples of how to implement the above problems in LINGO.

7.11.1 Solving a Nonlinear System in LINGO

To solve the system:

$$f_1(x,y) = x^2 + y^2 - 4 = 0 \quad , \, f_2(x,y) = x^2 - y - 1 = 0$$

LINGO Code:

```lindo
MODEL:
```

```
! Define variables
x = 1;  ! Initial guess for x
y = 1;  ! Initial guess for y
! Define equations
x2 + y2 = 4;
x2 - y = 1;
! Solve the system
@SOLVE();
END
```

7.11.2 Constrained Optimization in LINGO

To maximize the Cobb-Douglas production function:

$$Q(K,L) = K^{0.5}L^{0.5}$$

subject to:

$$2K + 3L = 100$$

LINGO Code:

```lindo
MODEL:
! Define variables
K = 1;  ! Initial guess for K
L = 1;  ! Initial guess for L
! Define objective function
MAX = K0.5  L0.5;
! Define constraint
```

```
  2K + 3L = 100;
  ! Solve the problem
  @SOLVE();
END
```

7.11.3 Utility Maximization in LINGO

To maximize the utility function:

$$U(x, y) = x^{0.4} y^{0.6}$$

subject to:

$$3x + 4y = 100$$

LINGO Code:

```lindo
MODEL:
  ! Define variables
  x = 1;  ! Initial guess for x
  y = 1;  ! Initial guess for y

  ! Define objective function
  MAX = x^0.4 y^0.6;
  ! Define constraint
  3x + 4y = 100;
  ! Solve the problem
  @SOLVE();
END
```

```

### 7.12 Tips for Using LINGO

1. Initial Guesses: Provide reasonable initial guesses for variables to ensure convergence.

2. Nonlinear Functions: Use parentheses to clarify the order of operations in nonlinear expressions.

3. Constraints: Ensure constraints are written in standard form (e.g., \[ g(x) \leq 0 \] or \[ g(x) = 0 \]).

4. Debugging: Use the `@SOLVE()` command to test small parts of your model before solving the entire problem.

***Conclusion:*** *the Jacobian method is a versatile tool for solving nonlinear systems and optimization problems. By combining theoretical understanding with practical implementation in tools like LINGO, you can efficiently solve complex problems in quantitative economics. The examples and exercises provided in this chapter will help you build a strong foundation in nonlinear programming.*

# Chapter 8: The Lagrange Method in Nonlinear Programming

### 8.1 Introduction to the Lagrange Method

The Lagrange method, named after the Italian-French mathematician Joseph-Louis Lagrange, is a fundamental technique in nonlinear programming for solving constrained optimization problems. It transforms a constrained problem into an unconstrained one by introducing additional variables known as Lagrange multipliers. This method is widely used in economics, engineering, and operations research to find optimal solutions subject to constraints.
```

In this chapter, we will explore the theoretical foundations of the Lagrange method, its applications in nonlinear programming, and its relevance to quantitative economics. We will also discuss practical examples, computational techniques, and implementation in optimization software like LINGO.

8.2 Theoretical Foundations

8.2.1 Constrained Optimization Problems

A general constrained optimization problem can be written as:

$$\max_{\mathbf{x}} f(\mathbf{x}) \quad \text{subject to} \quad g_i(\mathbf{x}) \le 0, \quad i = 1, 2, \ldots, m \; ; \; h_j(\mathbf{x}) = 0, \quad j = 1, 2, \ldots, p$$

where:

- $f(\mathbf{x})$ is the objective function to be maximized (or minimized),

- $g_i(\mathbf{x})$ are inequality constraints,

- $h_j(\mathbf{x})$ are equality constraints,

- $\mathbf{x}$ is the vector of decision variables.

8.2.2 The Lagrangian Function

The Lagrangian function combines the objective function and constraints into a single function using Lagrange multipliers:

$$\mathcal{L}(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = f(\mathbf{x}) - \sum_{i=1}^{m} \lambda_i g_i(\mathbf{x}) - \sum_{j=1}^{p} \mu_j h_j(\mathbf{x})$$

where:

- λ_i are the Lagrange multipliers for inequality constraints,

- μ_j are the Lagrange multipliers for equality constraints.

8.2.3 Karush-Kuhn-Tucker (KKT) Conditions

The KKT conditions are necessary for optimality in constrained optimization problems. For a solution $\mathbf{x}^*$ to be optimal, there must exist Lagrange multipliers λ_i^* and μ_j^* such that:

1. Stationarity:

$$\nabla f(\mathbf{x}^*) - \sum_{i=1} {}^*m\lambda_i {}^* \nabla g_i(\mathbf{x}^*) - \sum_{j=1} {}^*p\mu_j {}^* \nabla h_j(\mathbf{x}^*) = 0$$

.2. Primal Feasibility:

$$g_i(\mathbf{x}^*) \leq 0, \quad h_j(\mathbf{x}^*) = 0$$

3. Dual Feasibility:

$$\lambda_i^* \geq 0$$

4. Complementary Slackness:

$$\lambda_i {}^* g_i(\mathbf{x}^*) = 0$$

8.3 Applications in Quantitative Economics

8.3.1 Utility Maximization

Consider a consumer maximizing utility $U(x, y)$ subject to a budget constraint $p_x x + p_y y = I$. The Lagrangian is:

$$\mathcal{L}(x, y, \lambda) = U(x, y) - \lambda(p_x x + p_y y - I)$$

The first-order conditions are:

$$\frac{\partial \mathcal{L}}{\partial x} = \frac{\partial U}{\partial x} - \lambda p_x = 0 \frac{\partial \mathcal{L}}{\partial y} = \frac{\partial U}{\partial y} - \lambda p_y = 0 \frac{\partial \mathcal{L}}{\partial \lambda} = -(p_x x + p_y y - I) = 0$$

These conditions yield the optimal consumption bundle (x^*, y^*) and the marginal utility of income λ^*.

8.3.2 Production Cost Minimization

A firm minimizes cost $C(K, L) = rK + wL$ subject to producing a given output level $Q(K, L) = Q_0$. The Lagrangian is:

$$\mathcal{L}(K, L, \lambda) = rK + wL - \lambda(Q(K, L) - Q_0)$$

The first-order conditions are:

$$\frac{\partial \mathcal{L}}{\partial K} = r - \lambda \frac{\partial Q}{\partial K} = 0 \frac{\partial \mathcal{L}}{\partial L} = w - \lambda \frac{\partial Q}{\partial L} = 0 \frac{\partial \mathcal{L}}{\partial \lambda} = -(Q(K, L) - Q_0) = 0$$

These conditions yield the optimal input levels (K^*, L^*) and the shadow price of output λ^*.

8.4 Practical Examples

8.4.1 Example 1: Utility Maximization

Maximize the utility function:

$$U(x, y) = x^{0.5} y^{0.5}$$

subject to the budget constraint:

$$2x + 3y = 100$$

Solution:

1. Form the Lagrangian:

$$\mathcal{L}(x, y, \lambda) = x * 0.5 y * 0.5 - \lambda(2x + 3y - 100)$$

2. Compute the first-order conditions:

$$\frac{\partial \mathcal{L}}{\partial x} = 0.5x * -0.5y * 0.5 - 2\lambda = 0 \frac{\partial \mathcal{L}}{\partial y} = 0.5x * 0.5y * -0.5 - 3\lambda = 0 \frac{\partial \mathcal{L}}{\partial \lambda} = -(2x + 3y - 100) = 0$$

3. Solve the system to find $x^* = 25$, $y^* = 16.67$, and $\lambda^* = 0.083$.

8.4.2 Example 2: Cost Minimization

Minimize the cost function:

$$C(K, L) = 4K + 9L$$

subject to the production constraint:

$$Q(K, L) = K * 0.5L * 0.5 = 10$$

Solution:

1. Form the Lagrangian:

$$\mathcal{L}(K, L, \lambda) = 4K + 9L - \lambda(K * 0.5L * 0.5 - 10)$$

2. Compute the first-order conditions:

$$\frac{\partial \mathcal{L}}{\partial K} = 4 - 0.5\lambda K^{*-0.5} L^{*0.5} = 0 \frac{\partial \mathcal{L}}{\partial L} = 9 - 0.5\lambda K^{*0.5} L^{*-0.5} = 0 \frac{\partial \mathcal{L}}{\partial \lambda} = -(K^{*0.5} L^{*0.5} - 10) = 0$$

3. Solve the system to find $K^* = 22.5$, $L^* = 10$, and $\lambda^* = 12$.

8.5 Implementation in LINGO

8.5.1 Utility Maximization in LINGO

To maximize $U(x, y) = x^{*0.5} y^{*0.5}$ subject to $2x + 3y = 100$:

```lindo
MODEL:
  ! Define variables
  x = 1;  ! Initial guess for x
  y = 1;  ! Initial guess for y
  ! Define objective function
  MAX = x0.5  y0.5;
  ! Define constraint
  2x + 3y = 100;
  ! Solve the problem
  @SOLVE();
END
```

8.5.2 Cost Minimization in LINGO

To minimize $C(K, L) = 4K + 9L$ subject to $K^{*0.5} L^{*0.5} = 10$:

```lindo
MODEL:
```

```
  ! Define variables
  K = 1;  ! Initial guess for K
  L = 1;  ! Initial guess for L
  ! Define objective function
  MIN = 4K + 9L;
  ! Define constraint
  K0.5  L0.5 = 10;
  ! Solve the problem
  @SOLVE();
END
```

Conclusion: *the Lagrange method is a powerful tool for solving constrained optimization problems. By introducing Lagrange multipliers, it transforms complex constraints into manageable equations, enabling the use of calculus to find optimal solutions. Its applications in economics, such as utility maximization and cost minimization, highlight its importance in quantitative analysis.*

Conclusion

Throughout this course on Optimization Methods (Non-Linear Programming), we have explored a wide range of theoretical concepts, practical techniques, and computational tools essential for solving complex optimization problems. The course has provided a comprehensive foundation in both the mathematical principles and real-world applications of optimization, with a particular focus on non-linear programming and its relevance to quantitative economics.

Key Takeaways:

1. *Fundamentals of Optimization*: We began with an introduction to optimization, covering the basic components of optimization problems, including objective functions, decision variables, and constraints. We discussed various types of optimization problems, such as linear programming, non-linear programming, and convex optimization, and their applications in economics, engineering, and machine learning.

2. *Convexity and Its Role:* A significant portion of the course was dedicated to understanding convex sets and convex functions, which are crucial for ensuring that local optima are also global optima. We explored the properties of convex functions, the conditions for convexity, and how convexity simplifies optimization problems.

3. *Optimization Techniques:* We delved into several key optimization methods, including:

- *Newton's Method:* A second-order method that uses the Hessian matrix for fast convergence.

- *Steepest Ascent/Descent:* A first-order method that relies on the gradient to find the direction of the steepest increase or decrease.

- *Jacobian Method:* Used for solving systems of non-linear equations and optimizing functions with multiple variables.

- *Lagrange Method:* A powerful technique for solving constrained optimization problems by introducing Lagrange multipliers.

4. *Applications in Economics:* The course emphasized the practical applications of optimization in quantitative economics, such as utility maximization, cost minimization, and

portfolio optimization. These applications demonstrated how optimization techniques can be used to model and solve real-world economic problems.

5. *Computational Tools:* We explored the use of computational tools like R, Lindo, and QM4Windows to implement and solve optimization problems. These tools are essential for handling complex problems that require numerical solutions.

6. *Problem-Solving Skills:* Through numerous exercises and examples, the course equipped students with the skills to formulate, analyze, and solve optimization problems. The exercises ranged from simple graphical methods to more advanced techniques like gradient descent and the simplex method.

Relevance to Quantitative Economics:

Optimization methods are indispensable in quantitative economics, where decision-making often involves maximizing or minimizing an objective function subject to constraints. Whether it's maximizing utility, minimizing costs, or optimizing resource allocation, the techniques covered in this course provide the necessary tools to tackle these challenges effectively. The ability to apply these methods using computational tools further enhances their practical utility in economic analysis and policy-making.

As optimization continues to play a critical role in various fields, including machine learning, operations research, and engineering, the knowledge gained from this course will serve as a strong foundation for further exploration. Students are encouraged to delve deeper into advanced topics such as stochastic optimization, dynamic programming, and global optimization techniques. Additionally, staying updated with the latest developments in optimization software and algorithms will be beneficial for both academic and professional growth.

This course has not only provided a solid theoretical understanding of optimization methods but also emphasized their practical application through hands-on exercises and case studies. By mastering these techniques, students are now well-equipped to approach complex optimization problems with confidence, whether in academic research, economic modeling, or real-world decision-making.

Dr. Fatih CHELLAI

Reference

Author(s)	Year	Title	Publisher	Relevance
Boyd, S., & Vandenberghe, L.	2004	*Convex Optimization*	Cambridge University Press	Definitive text on convex optimization, covering convex sets, functions, and techniques.
Nocedal, J., & Wright, S. J.	2006	*Numerical Optimization*	Springer	Comprehensive resource on numerical optimization methods, including gradient-based and Newton's methods.
Luenberger, D. G., & Ye, Y.	2016	*Linear and Nonlinear Programming*	Springer	Thorough introduction to linear and non-linear programming, with a focus

				on practical algorithms.
Chong, E. K. P., & Żak, S. H.	2013	*An Introduction to Optimization*	Wiley	Accessible introduction to optimization techniques, suitable for beginners and advanced learners.
Bazaraa, M. S., Sherali, H. D., & Shetty, C. M.	2006	*Nonlinear Programming: Theory and Algorithms*	Wiley	Classic text on non-linear programming, strong on Jacobian method and Lagrange multipliers.
Varian, H. R.	1992	*Microeconomic Analysis*	W.W. Norton & Company	Standard reference for microeconomic theory, with detailed discussions of optimization techniques.
Mas-Colell, A., Whinston, M. D., & Green, J. R.	1995	*Microeconomic Theory*	Oxford University Press	Comprehensive text on microeconomic theory, with extensive coverage of optimization methods.
Simon, C. P., & Blume, L.	1994	*Mathematics for Economists*	W.W. Norton & Company	Provides a solid mathematical foundation for economists, including optimization techniques.
R Core Team	2023	*R: A Language and Environment for Statistical Computing*	R Foundation for Statistical Computing	Official documentation for R, widely used for implementing optimization algorithms.
LINDO Systems	2023	*LINGO User's Guide*	LINDO Systems	Official user guide for LINGO, useful for solving linear, non-linear, and integer optimization problems.
QM4Windows	2023	*Quantitative Methods for Windows User Manual*	QM4Windows	Guide to using QM4Windows, a tool for solving optimization problems.
Bertsekas, D. P.	1999	*Nonlinear Programming*	Athena Scientific	Advanced text on non-linear programming, focusing on

				theoretical and algorithmic aspects.
Fletcher, R.	2013	*Practical Methods of Optimization*	Wiley	Practical insights into optimization methods, including gradient-based and quasi-Newton methods.
Rockafellar, R. T.	1970	*Convex Analysis*	Princeton University Press	Classic text on convex analysis, providing a rigorous mathematical treatment of convex sets and functions.