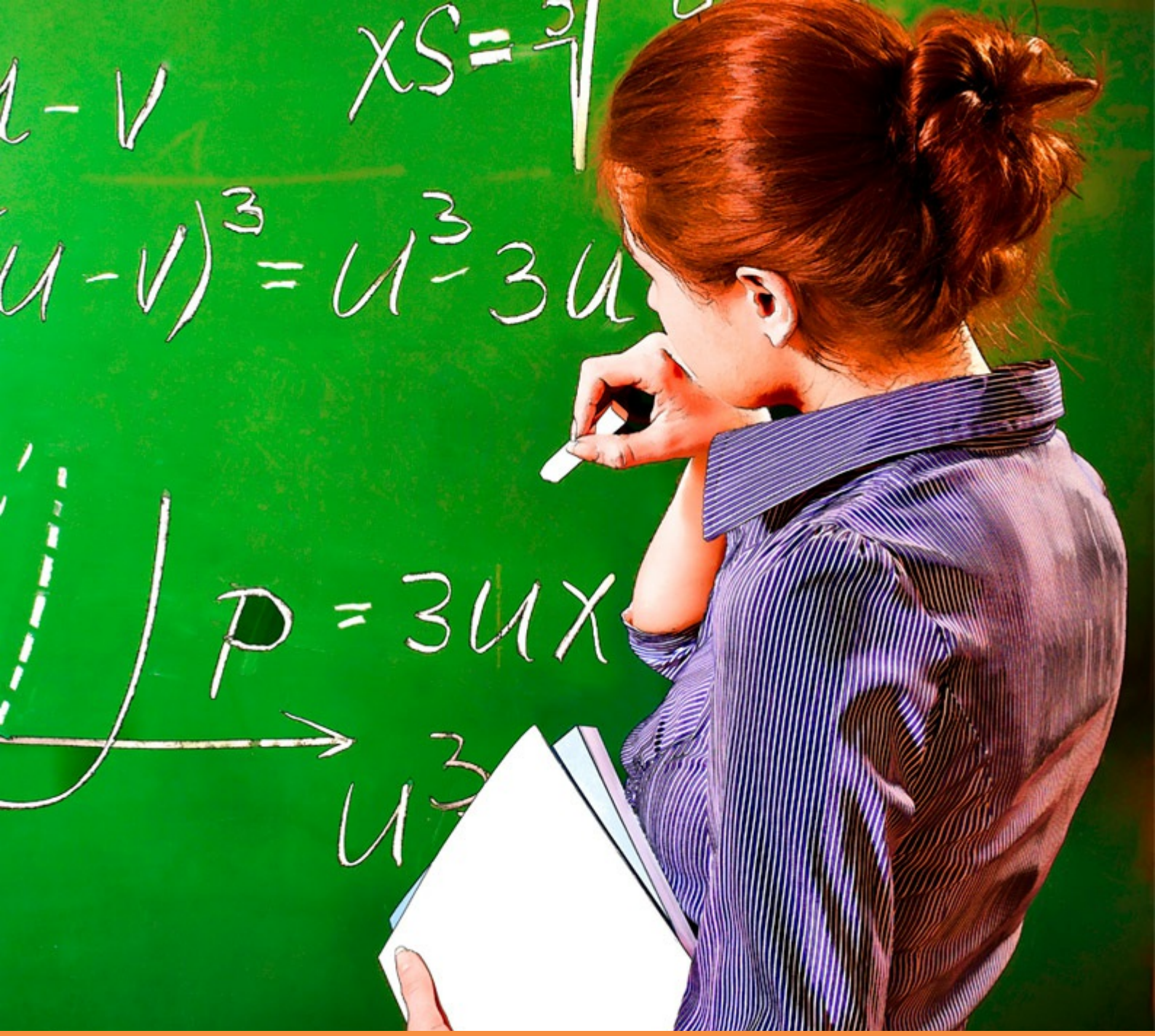


Fundamental Engineering Optimization Methods

Kamran Iqbal

bookboon
The eBook company

Kamran Iqbal

Fundamental Engineering Optimization Methods

Download free eBooks at bookboon.com

Fundamental Engineering Optimization Methods
1st edition
© 2013 Kamran Iqbal & bookboon.com
ISBN 978-87-403-0489-3

Download free eBooks at bookboon.com

Contents

Download free eBooks at bookboon.com

Download free eBooks at bookboon.com

Download free eBooks at bookboon.com

Download free eBooks at bookboon.com

Preface

This book is addressed to students in fields of engineering and technology as well as practicing engineers. It covers the fundamentals of commonly used optimization methods used in engineering design. Optimization methods fall among the mathematical tools typically used to solve engineering problems. It is therefore desirable that graduating students and practicing engineers are equipped with these tools and are trained to apply them to specific problems encountered in engineering practice.

Optimization is an integral part of the engineering design process. It focuses on discovering optimum solutions to a design problem through systematic consideration of alternatives, while satisfying resource and cost constraints. Many engineering problems are open-ended and complex. The overall design objective in these problems may be to minimize cost, to maximize profit, to streamline production, to increase process efficiency, etc. Finding an optimum solution requires a careful consideration of several alternatives that are often compared on multiple criteria.

Mathematically, the engineering design optimization problem is formulated by identifying a cost function of several optimization variables whose optimal combination results in the minimal cost. The resource and other constraints are similarly translated into mathematical relations. Once the cost function and the constraints have been correctly formulated, analytical, computational, or graphical methods may be employed to find an optimum. The challenge in complex optimization problems is finding a global minimum, which may be elusive due to the complexity and nonlinearity of the problem.

This book covers the fundamentals of optimization methods for solving engineering problems. Written by an engineer, it introduces fundamentals of mathematical optimization methods in a manner that engineers can easily understand. The treatment of the topics presented here is both selective and concise. The material is presented roughly at senior undergraduate level. Readers are expected to have familiarity with linear algebra and multivariable calculus. Background material has been reviewed in Chapter 2.

The methods covered in this book include: a) analytical methods that are based on calculus of variations; b) graphical methods that are useful when minimizing functions involving a small number of variables; and c) iterative methods that are computer friendly, yet require a good understanding of the problem. Both linear and nonlinear methods are covered. Where necessary, engineering examples have been used to build an understanding of how these methods can be applied. Though not written as text, it may be used as text if supplemented by additional reading and exercise problems from the references.

Download free eBooks at bookboon.com

There are many good references available on the topic of optimization methods. A short list of prominent books and internet resources appears in the reference section. The following references are main sources for this manuscript and the topics covered therein: Arora (2012); Belegundu and Chandrupatla (2012); Chong and Zak (2013); and, Griva, Nash & Sofer (2009). In addition, lecture notes of eminent professors who have regularly taught optimization classes are available on the internet. For details, the interested reader may refer to these references or other web resources on the topic.

Download free eBooks at bookboon.com

1 Engineering Design Optimization

This chapter introduces the topic of optimization through example problems that have been selected from various fields including mathematics, economics, computer science, and engineering.

Learning Objectives: The learning goal in this chapter is to develop an appreciation for the topic as well as the diversity and usefulness of the mathematical and computational optimization techniques.

1.1 Introduction

Engineering system design comprises selecting one or more variables to meet a set of objectives. A better design is obtained if an appropriate cost function can be reduced. The design is optimum when the cost is the lowest among all feasible designs. Almost always, the design choices are limited due to resource constraints, such as material and labor constraints, as well as physical and other restrictions. A feasible region in the design space is circumscribed by the constraint boundaries. More importantly, both the cost function and the constraints can be cast as mathematical functions involving design variables. The resulting mathematical optimization problem can then be solved using methods discussed in this book.

Engineering system design is an interdisciplinary process that necessitates cooperation among designers from various engineering fields. Engineering design can be a complex process. It requires assumptions to be made to develop models that can be subjected to analysis and verification by experiments. The design of a system begins by analyzing various options. For most applications the entire design project must be broken down into several subproblems which are then treated independently. Each of the subproblems can be posed as an optimum design problem to be solved via mathematical optimization.

A typical optimum engineering design problem may include the following steps: a descriptive problem statement, preliminary investigation and data collection as a prelude to problem formulation, identification of design variables, optimization criteria and constraints, mathematical formulation of the optimization problem, and finding a solution to the problem. This text discusses the last two steps in the design process, namely mathematical formulation and methods to solve the design optimization problem.

Engineering design optimization is an open-ended problem. Perhaps the most important step toward solving the problem involves correct mathematical formulation of the problem. Once the problem has been mathematically formulated, analytical and computer methods are available to find a solution. Numerical techniques to solve the mathematical optimization problems are collectively referred as mathematical programming framework. The framework provides a general and flexible formulation for solving engineering design problems.

Download free eBooks at bookboon.com

Some mathematical optimization problems may not have a solution. This usually happens due to conflicting requirements of incorrect formulation of the optimization problem. For example, constraints may be restrictive so that no feasible region can be found, or the feasible region may be unbounded due to a missing constraint. In this text we will assume that the problem has been correctly formulated so that the feasible region is closed and bounded.

1.2 Optimization Examples in Science and Engineering

We wish to introduce the topic of optimization with the help of examples. These examples have been selected from various STEM (science, technology, engineering, mathematics) fields. Each example requires finding the optimal values of a set of design variables in order to optimize (maximize or minimize) a generalized cost that may represent the manufacturing cost, profit, energy, power, distance, mean square error, and so on. The complexity of the design problem grows with number of variables involved. Each of the simpler problems, presented first, involves a limited number of design variables. The problems that follow are more complex in nature and may involve hundreds of design variables. Mathematical formulation of each problem is provided following the problem definition. While the simpler problems are relatively easy to solve by hand, the complex problems require the use of specialized optimization software in order to find a solution.

Problem 1: Student diet problem

A student has a choice of breakfast menu (eggs, cereals, tarts) and a limited ($10) budget to fulfill his/her nutrition needs (1000 calories, 100 g protein) at minimum cost. Eggs provide 500 calories and 50g protein and cost $3.50; cereals provide 500 calories and 40g protein and cost $3; tarts provide 600 calories and 20g protein and cost $2. How does he/she choose his/her breakfast mix?

Formulation: Let $x^T = [x_1, x_2, x_3]$ represent the quantities of eggs, cereals and tarts chosen for breakfast. Then, the optimization problem is mathematically formulated as follows:

$$\begin{aligned} &\min_{x_1,x_2,x_3} f = 3.5x_1 + 3x_2 + 2x_3 \\ &\text{Subject to: } 500(x_1 + x_2) + 600x_3 \geq 1000,\ 50x_1 + 40x_2 + 20x_3 \geq 100, \\ &\qquad 3.5x_1 + 3x_2 + 2x_3 \leq 10;\ x_1, x_2, x_3 \in \mathbb{Z} \end{aligned} \tag{1.1}$$

Problem 2: Simplified manufacturing problem

A manufacturer produces two products: tables and chairs. Each table requires 10 kg of material and 5 units of labor, and earns $7.50 in profit. Each chair requires 5 kg of material and 12 units of labor, and earns $5 in profit. A total of 60 kg of material and 80 units of labor are available. Find the best production mix to earn *maximum profit*.

Download free eBooks at bookboon.com

Formulation: Let Let $x^T = [x_1, x_2]$ represent the quantities of tables and chairs to be manufactured. Then, the optimization problem is mathematically formulated as follows:

$$\begin{aligned} &\max_{x_1,x_2} f = 7.5x_1 + 5x_2 \\ &\text{Subject to: } 10x_1 + 5x_2 \leq 60, 5x_1 + 12x_2 \leq 80; x_1, x_2 \in \mathbb{Z} \end{aligned} \tag{1.2}$$

Problem 3: Shortest distance problem

Find the *shortest distance* from a given point (x_0, y_0) to a given curve: $y = f(x)$.

Formulation: The optimization problem is mathematically formulated to minimize the Euclidian distance from the given point to the curve:

$$\begin{aligned} &\min_{x,y} f = \frac{1}{2}\{(x - x_0)^2 + (y - y_0)^2\} \\ &\text{Subject to: } y = f(x) \end{aligned} \tag{1.3}$$

Problem 4: Data-fitting problem

Given a set of N data points $(x_i, y_i), i = 1, \ldots, N$, fit a polynomial of degree to the data such that the *mean square error* $\sum_{i=1}^{N}\big(y_i - f(x_i)\big)^2$ is *minimized.*

Formulation: Let the polynomial be given as: $y = f(x) = a_0 + a_1 x + \cdots + a_m x^m$; then, the unconstrained optimization problem is formulated as:

$$\min_{a_0,a_1} f = \frac{1}{2}\sum_{i=1}^{N}(y_i - a_0 - a_1 x_i - \cdots - a_m x_i^m)^2 \tag{1.4}$$

Problem 5: Soda can design problem

Design a soda can (choose diameter *d* and height *h*) to hold a volume of 200 ml, such that the *manufacturing cost* (a function of surface area) *is minimized* and the constraint $h \geq 2d$ is obeyed.

Formulation: Let $\boldsymbol{x}^T = [d, l]$ represent the diameter and length of the can. Then, the optimization problem is formulated to minimize the surface area of the can as:

$$\begin{aligned} &\min_{d,l} f = \frac{1}{4}\pi d^2 + \pi d l \\ &\text{Subject to: } \tfrac{1}{4}\pi d^2 l = 200, 2d - h \leq 0 \end{aligned} \tag{1.5}$$

Problem 6: Open box problem

What is the *largest volume* for an open box that can be constructed from a given sheet of paper (8.5″×11″) by cutting out squares at the corners and folding the sides?

Download free eBooks at bookboon.com

Formulation: Let x represent the side of the squares to be cut; then, the unconstrained optimization problem is formulated as:

$$\max_{x} f = x(8.5 - 2x)(11 - 2x) \tag{1.6}$$

Problem 7: Ladder placement problem

What are the dimensions (width, height) of the *largest box* that can be placed under a ladder of length l when the ladder rests against a vertical wall?

Formulation: Let $[x, y]$ represent the dimensions of the box, and let $(a, 0)$ and $(0, b)$ represent the horizontal and vertical contact points of the ladder with the floor and the wall, respectively. Then, the optimization problem is mathematically formulated as:

$$\max_{x,y} f = xy$$
$$\text{Subject to: } \frac{x}{a} + \frac{y}{b} \leq 1, a^2 + b^2 = l \tag{1.7}$$

Problem 8: Logging problem

What are the dimensions of a rectangular beam of *maximum dimensions* (or volume) that can be cut from a log of given dimensions?

Download free eBooks at bookboon.com

Formulation: Let $[x, y]$ represent the width and height of the beam to be cut, and let d represent the diameter of the log. Then, the optimization problem is formulated as:

$$\max_{xy} f = xy$$
$$\text{Subject to: } x^2 + y^2 - d^2 \leq 0 \tag{1.8}$$

Problem 9: Knapsack problem

Given an assortment of n items, where each item i has a value $c_i > 0$, and a weight $w_i > 0$, fill a knapsack of given capacity (weight W) so as to *maximize the value of the included items.*

Formulation: Without loss of generality, we assume that $W = 1$. Let $x_i \in \{0,1\}$. Let denote the event that item *i* is selected for inclusion in the sack; then, the knapsack problem is formulated as:

$$\max_{x_i} f = \sum_{i=1}^{n} c_i x_i$$
$$\text{Subject to: } \sum_{i=1}^{n} w_i x_i \leq 1 \tag{1.9}$$

Problem 10: Investment problem

Given the stock prices p_i and anticipated rates of return r_i associated with a group of investments, choose a mix of securities to invest a sum of $1M in order to *maximize return on* investment.

Formulation: Let $x_i \in \{0,1\}$ express the inclusion of security *i* in the mix, then the investment problem is modeled as the knapsack problem (Problem 9).

Problem 11: Set covering problem

Given a set $S = \{e_i : i = 1, \dots, m\}$ and a collection $\mathcal{S} = \{S_j : j = 1, \dots, n\}$ of subsets of S, with associated costs c_j, find the smallest sub-collection Σ of $\mathcal{S}$ that covers i.e., S, i.e., $\bigcup_{S_j \in \Sigma} S_j = S$.

Formulation: Let $a_{ij} \in \{0,1\}$ denote the condition that $e_i \in S_j$, and let $x_j \in \{0,1\}$ and let denote the condition that $S_j \in \Sigma$; then, the set covering problem is formulated as:

$$\max_{x_j} f = \sum_{j=1}^{n} c_j x_j$$
$$\text{Subject to: } \sum_{j=1}^{n} a_{ij} x_i \geq 1 \,, i = 1, \dots, m; x_j \in \{0,1\}, j = 1, \dots, n \tag{1.10}$$

Problem 12: Airline scheduling problem

Given the fixed costs and operating costs per segment, design an optimum flight schedule to *minimize total operating cost* for given passenger demand on each segment over a network of routes to be serviced under given connectivity, compatibility, and resource (aircrafts, manpower) availability constraints.

Download free eBooks at bookboon.com

Formulation: Let $S = \{e_i: i = 1, \dots, m\}$ denote the set of flight segments required to be covered, and let each subset $S_j \subseteq S$ denote a set of connected flight segments that can be covered by an aircraft or a crew; then the least cost problem to cover the available routes can be formulated as a set covering problem (Problem 10).

Problem 13: Shortest path problem

Find the *shortest path* from *node p to node q in a connected graph* (*V, E*), *where V denotes the vertices and denotes the edges.*

Formulation: Let e_{ij} denote the edge incident to both nodes i and j, and let $f: E \to \mathbb{R}$ represent a real-valued weight function; further, let $P = (v_1, v_2, \dots, v_n)$ denote a path, where $v_1 = p, v_n = q$; then, the unconstrained single-pair shortest path problem is formulated as:

$$\min_n f = \sum_{i=1}^{n-1} e_{i,i+1} \tag{1.11}$$

Alternatively, let x_{ij} denote a variable associated with e_{ij}; then, an integer programming formulation (Chapter 6) of the shortest path problem is given as:

$$\min_{x_{ij}} f = \sum_{i,j} e_{ij} x_{ij}$$
$$\text{Subject to: } \sum_j x_{ij} - \sum_j x_{ji} = \begin{cases} 1 \text{ for } i = p \\ -1 \text{ for } i = q \\ 0 \text{ otherwise} \end{cases} \tag{1.12}$$

Note: the shortest path problem is a well-known problem in graph theory and algorithms, such as Dijkstra's algorithm or Bellman-Ford algorithm, are available to solve variants of the problem.

Problem 14: Traveling salesman problem

A company requires a salesman to visit its *N* stores (say 50 stores) that are geographically distributed in different locations. Find the visiting sequence that will require the *least amount of overall travel.*

Formulation: The traveling salesman problem is formulated as shortest path problem in an undirected weighted graph where the stores represent the vertices of the graph. The problem is then similar to Problem 10.

Download free eBooks at bookboon.com

Problem 15: Transportation problem

Goods are to be shipped from m supply points with capacities: $s_1, s_2, \dots, s_m$, to distribution points with demands: $d_1, d_2, \dots, d_n$. Given the transportation cost c_{ij} for each of the network routes, find the optimum quantities, x_{ij}, to be shipped along those routes to *minimize total shipment cost.*

Formulation: let x_{ij}, denote the quantity to be shipped node *i* to node *j*; then, the optimization problem is formulated as:

$$\begin{aligned} &\min_{x_{ij}} f = \sum_{i,j} c_{ij} x_{ij} \\ &\text{Subject to: } \sum_{j} x_{ij} = s_i, \text{for } i = 1, \dots, m; \ \sum_{i} x_{ij} = d_j, \text{for } i = 1, \dots, n; x_{ij} \geq 0 \end{aligned} \tag{1.13}$$

Problem 16: Power grid estimation problem

Given the measurements of active and reactive power flows (p_{ij}, q_{ij}) between nodes *i, j* and the measurements v_i of the node voltages in an electric grid, obtain the *best estimate* of the state of the grid, i.e., solve for complex node voltages: $\mathrm{v}_i = v_i \angle \delta_i$, where δ_i represents the phase angle.

Download free eBooks at bookboon.com

Formulation: let $\bar{v}_i, \bar{p}_{ij}, \bar{q}_{ij}$ represent the measured variables, and $k_i^v, k_{ij}^p, k_{ij}^q$, let respectively, represent the confidence in measurements of the node voltages and the power flows; further let $z_{ij} = z_{ij}\angle\theta_{ij}$ represent the complex impedance between nodes i, j; then, power grid state estimation problem is formulated as (Pedregal, p. 11):

$$\min_{v_i,\delta_i} f = \sum_i k_i^v (v_i - \bar{v}_i)^2 + \sum_{i,j} k_{ij}^p \left(p_{ij} - \bar{p}_{ij}\right)^2 + \sum_{i,j} k_{ij}^q \left(q_{ij} - \bar{q}_{ij}\right)^2$$
$$\text{Subject to: } \begin{cases} p_{ij} = \frac{v_i^2}{z_{ij}} \cos\theta_{ij} - \frac{v_i v_j}{z_{ij}} \cos\left(\theta_{ij} + \delta_i - \delta_j\right) \\ q_{ij} = \frac{v_i^2}{z_{ij}} \sin\theta_{ij} - \frac{v_i v_j}{z_{ij}} \sin(\theta_{ij} + \delta_i - \delta_j) \end{cases} \tag{1.14}$$

Problem 17 Classification problem

Given a set of data points: $\boldsymbol{x}_i \in \mathbb{R}^n, i = 1, \ldots, n$, with two classification labels: $y_i \in \{1, -1\}$, find the equation of a hyperplane separating data into classes with *maximum inter-class distance*.

Formulation: To simplify the problem, we assume that data points lie in a plane, i.e., $\boldsymbol{x}_i \in \mathbb{R}^2$, and that they are linearly separable. We consider a hyperplane of the form: $\boldsymbol{w}^T\boldsymbol{x} - b = 0$, where $\boldsymbol{w}$ is a weight vector that is normal to the hyperplane. For separating given data points, we assume that $\boldsymbol{w}^T\boldsymbol{x}_i - b \geq 1$ for points labeled as 1, and $\boldsymbol{w}^T\boldsymbol{x}_i - b \leq -1$ for points labeled as –1. The two hyperplanes (lines) are separated by $\frac{2}{\|\boldsymbol{w}\|}$. Thus, optimization problem is defined as:

$$\max_{\boldsymbol{w}} \frac{1}{2}\|\boldsymbol{w}\|^2$$
$$\text{Subject to: } 1 - y_i(\boldsymbol{w}^T\boldsymbol{x}_i - b) \leq 0; i = 1, \ldots, n \tag{1.15}$$

Problem 18: Steady-state finite element analysis problem

Find nodal displacements u_i that *minimize the total potential energy* associated with a set of point masses m_i connected via springs of constants k_{ij}, while obeying structural and load constraints.

Formulation: For simplicity we consider a one-dimensional version of the problem, where the nodal displacements are represented as: $u_1, u_2, \ldots, u_N$. Let f_i represent an applied force at node i; then, the potential energy minimization problem is formulated as:

$$\min_{u_i} \prod = \frac{1}{2}\sum_{i,j} k_{ij} u_i u_j + \sum_i u_i f_i \tag{1.16}$$

Download free eBooks at bookboon.com

Problem 19: Optimal control problem

Find an admissible control sequence $u(t)$that *minimizes a quadratic cost function* $J(x,u,t)$, while moving a dynamic system: $\dot{x} = Ax + Bu$ between prescribed end points. The class of optimal control problems includes minimum *energy* and minimum *time* problems, among others.

Formulation: As a simplified problem, we consider the optimal control of an inertial system of unit mass modeled with position (x) and velocity (v). The system dynamics are given as: $\dot{x} = v, \dot{v} = u$, where $u(t), t \in [0,T]$ represents the input. We consider a quadratic cost that includes time integral of square of position and input variables. The resulting optimal control problem is formulated as:

$$\begin{aligned} &\min_{x_i} f = \int_0^T \tfrac{1}{2}(x^2 + \rho u^2)dt \\ &\text{Subject to: } \dot{x} = v, \dot{v} = u \end{aligned} \tag{1.17}$$

1.3 Notation

The following notation is used throughout this book: $\mathbb{R}$ denotes the set of real numbers; $\mathbb{R}^n$ denotes the set of real n-vectors; $\mathbb{R}^{m\times n}$ denotes the set of real $m \times n$ matrices; $f: \mathbb{R}^n \to \mathbb{R}^m$ denotes an $\mathbb{R}^m$-valued function defined over ; $\mathbb{R}^n$; $\mathbb{Z}$ denotes the set of integers, and $\mathbb{Z}^n$ denotes integer vectors. In the text, small bold face letters such as $\boldsymbol{x}, \boldsymbol{y}$ are used to represent vectors or points in $\mathbb{R}^n$; capital bold face letters such $\boldsymbol{A}, \boldsymbol{B}$ as are used to represent matrices; $\boldsymbol{A}_q$ represents qth column of $\boldsymbol{A}$; and $\boldsymbol{I}$ represents an identity matrix.

Download free eBooks at bookboon.com

2 Mathematical Preliminaries

This chapter introduces essential mathematical concepts that are required to understand the material presented in later chapters. The treatment of the topics is concise and limited to presentation of key aspects of the topic. More details on these topics can be found in standard mathematical optimization texts. Interested readers should consult the references (e.g., Griva, Nash & Sofer, 2009) for details.

Learning Objectives: The learning goal in this chapter is to understand the mathematical principles necessary for formulating and solving optimization problems, i.e., for understanding the optimization techniques presented in later chapters.

2.1 Set Definitions

Closed Set. A set is closed if for any sequence of points $\{x_k\}$, $x_k \in S$, $\lim_{k\to\infty} x_k = x$, we have $x \in S$. For example, the set $S = \{x: |x| \leq c\}$ where c is a finite number, describes a closed set.

Bounded Set. A set S is bounded if for every $x \in S, \|x\| < c$, where $\|\cdot\|$ represents a vector norm and c is a finite number.

Compact set. A set S is compact if it is both closed and bounded.

Download free eBooks at bookboon.com

Interior point. A point is $x \in S$ interior to the set if $\{y: \|y - x\| < \epsilon\} \subset S$ for some $\epsilon > 0$.

Open Set. A set *S* is open if every $x \in S$ is an interior point of *S*. For example, the set $S = \{x: |x| < c\}$, where *c* is a finite number, is an open set.

Convex Set. A set *S* is convex if for each pair $x, y \in S$, their convex combination $\alpha x + (1 - \alpha)y \in S$ for $0 \leq \alpha \leq 1$. Examples of convex sets include a single point, a line segment, a hyperplane, a halfspace, the set of real numbers $(\mathbb{R})$, and $\mathbb{R}^n$.

Hyperplane. The set $S = \{\boldsymbol{x}: \boldsymbol{a}^T\boldsymbol{x} = b\}$, where $\boldsymbol{a}$ and b are constants defines a hyperplane. Note that in two dimensions a hyperplane is a line. Also, note that vector $\boldsymbol{a}$ is normal to the hyperplane.

Halfspace. The set $S = \{\boldsymbol{x}: \boldsymbol{a}^T\boldsymbol{x} \leq b\}$, where $\boldsymbol{a}$ and b are constants defines a halfspace. Note that vector $\boldsymbol{a}$ is normal to the halfspace. Also, note that a halfspace is convex.

Polyhedron. A polyhedron represents a finite intersection of hyperplanes and halfspaces. Note that a polyhedron is convex.

Convex Hull. The convex hull of a set *S* is the set of all convex combinations of points in *S*. Note that convex hull of *S* is the smallest convex set that contains *S*.

Extreme Point. A point $x \in S$ is an extreme point (or vertex) of a convex *S* set if it cannot be expressed as $x = \alpha y + (1 - \alpha)z$, with $y, z \in S$ where $y, z \neq x$, and $0 < \alpha < 1$.

2.2 Function Definitions

Function. A function $f(\boldsymbol{x})$ describes a mapping from a set of points called domain to a set of points called range. Mathematically, $f: \mathcal{D} \to \mathcal{R}$ where $\mathcal{D}$ denotes the domain and $\mathcal{R}$ the range of the function.

Continuous Function. A function $f(\boldsymbol{x})$ is said to be continuous at a point $\boldsymbol{x}_0$ if $\lim_{\boldsymbol{x} \to \boldsymbol{x}_0} f(\boldsymbol{x}) = f(\boldsymbol{x}_0)$. Alternatively, if a sequence of points $\{\boldsymbol{x}_k\}$ in the function domain $\mathcal{D}(f)$ converges to $\boldsymbol{x}_0$, then $f(\boldsymbol{x}_k)$ must converge to $f(\boldsymbol{x}_0)$ for a function to be continuous. Note, that for functions of single variable, this implies that left and right limits coincide.

Affine Function. A function of the form $f(\boldsymbol{x}) = \boldsymbol{a}^T\boldsymbol{x} + b$ of the form represents an affine function.

Quadratic Function. A function of the form $f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^T\boldsymbol{Q}\boldsymbol{x} - \boldsymbol{b}^T\boldsymbol{x}$, where $\boldsymbol{Q}$ is symmetric, represents a quadratic function.

Download free eBooks at bookboon.com

Level Sets. The level sets of a function are defined as $S = \{x: f(\boldsymbol{x}) = c\}$. For functions of a single variable, the level sets represent discrete points. For functions of two variables, level sets are contours plotted in the xy- plane.

Stationary Point. From elementary calculus, a single-variable function $f(\boldsymbol{x})$ has a stationary point at x_0 if the derivative $f'(x)$ vanishes at x_0, i.e., $f'(x_0) = 0$. Graphically, the slope of the function is zero at the stationary point, which may represent a minimum, a maximum, or a point of inflecion.

Local Minimum. A multi-variable function, $f(\boldsymbol{x})$, has a local minimum at $\boldsymbol{x}^*$ if $f(\boldsymbol{x}^*) \leq f(\boldsymbol{x})$ in a small neighborhood around $\boldsymbol{x}^*$, defined by $|\boldsymbol{x} - \boldsymbol{x}^*| < \epsilon$.

Global Minimum. The multi-variable function $f(\boldsymbol{x})$ has a global minimum at $\boldsymbol{x}^*$ if $f(\boldsymbol{x}^*) \leq f(\boldsymbol{x})$ for all $\boldsymbol{x}$ in a feasible region defined by the problem.

Convex Functions. A function $f(x)$ defined on a convex set S is convex if and only if for all $\boldsymbol{x}, \boldsymbol{y} \in S$, $f(\alpha\boldsymbol{x} + (1-\alpha)\boldsymbol{y}) \leq \alpha f(\boldsymbol{x}) + (1-\alpha) f(\boldsymbol{y})$, $\alpha \in [0,1]$. Note that affine functions defined over convex sets are convex. Similarly, quadratic functions defined over convex sets are convex.

2.3 Taylor Series Approximation

Taylor series approximates a differentiable function $f(x)$ in the vicinity of an operating point x_0. Such approximation is helpful in several problems involving functions.

An infinite Taylor series expansion of $f(x)$ around x_0 (where $d = x - x_0$) is given as:

$$f(x_0 + d) = f(x_0) + f'(x_0)d + \frac{1}{2!}f''(x_0)d^2 + \cdots$$

As an example, the Taylor series for sin and cosine functions around $x_0 = 0$ are given as:

$$\sin x = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \cdots$$
$$\cos x = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \cdots$$

These series are summed in the Euler formula: $\cos x + i \sin x = e^{-ix}$.

The nth order Taylor series approximation of $f(x)$ is given as:

$$f(x_0 + d) \cong f(x_0) + f'(x_0)d + \frac{1}{2!}f''(x_0)d^2 + \cdots + \frac{1}{n!}f^{(n)}(x_0)d^n$$

Download free eBooks at bookboon.com

We note that first or second order approximation often suffice in the close neighborhood of x_0. As an example, the local behavior of a function is frequently approximated by a tangent line defined as:

$$f(x) - f(x_0) \cong f'(x_0)(x - x_0)$$

Next, the Taylor series expansion of a function $f(x_1, x_2)$ of two variables at a point (x_{10}, x_{20}) is given as:

$$f(x_1 + d_1, x_2 + d_2) = f(x_{10}, x_{20}) + \frac{\partial f}{\partial x_1} d_1 + \frac{\partial f}{\partial x_2} d_2 + \frac{1}{2}\left[\frac{\partial^2 f}{\partial x_1^2} d_1^2 + \frac{\partial^2 f}{\partial x_1 \partial x_2} \mathrm{d}_1 \mathrm{d}_2 + \frac{\partial^2 f}{\partial x_2^2} d_2^2\right] + \cdots$$

where $d_1 = x_1 - x_{10}, d_2 = x_2 - x_{20}$, and all partial derivatives are computed at the point: (x_{10}, x_{20}). Further, let $z = f(x_1, x_2)$;then, the tangent plane of z at (x_{10}, x_{20}) is defined by the equation:

$$z = f(x_{10}, x_{20}) + \frac{\partial f}{\partial x_1}\Big|_{(x_{10}, x_{20})} (x_1 - x_{10}) + \frac{\partial f}{\partial x_2}\Big|_{(x_{10}, x_{20})} (x_2 - x_{20}).$$

Taylor series expansion in the case of a multi-variable function is given after defining gradient vector and Hessian matrix in Sec. 2.4. Finally, it is important to remember that Taylor series only approximates the local behavior of the function, and therefore should be used with caution.

Download free eBooks at bookboon.com

2.4 Gradient Vector and Hessian Matrix

The gradient vector and Hessian matrix play an important role in optimization. These concepts are introduced as such:

The Gradient Vector. Let $f(\boldsymbol{x}) = f(x_1, x_2, \dots, x_n)$ be a real-valued function of variables with continuous partial derivatives, i.e., $f \in \mathrm{C}^1$.. Then, the gradient of f is a vector defined by:

$$\nabla f(\boldsymbol{x})^T = \left(\frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}, \dots, \frac{\partial f}{\partial x_n}\right)$$

The gradient vector has several important properties. These include:

1. The gradient points in the direction of maximum rate of increase in the function value at a given point. This can be seen by considering the directional derivative of $f(\boldsymbol{x})$ along any direction $\boldsymbol{d}$, defined as: $f_{\boldsymbol{d}}{}'(\boldsymbol{x}) = \nabla f(\boldsymbol{x})^T \boldsymbol{d} = |\nabla f(\boldsymbol{x})||\boldsymbol{d}| \cos\theta$, where θ is the angle between the two vectors. Then, the maximum rate of increase occurs when $\theta = 0$, i.e., along $\nabla f(\boldsymbol{x})$.
2. The magnitude of the gradient gives the maximum rate of increase in $f(\boldsymbol{x})$. Indeed, $\max_{\|d\|=1} f_d{}'(x) = \|\nabla f(\boldsymbol{x})\|$.
3. The gradient vector at a point $\boldsymbol{x}^*$ is normal to the tangent hyperplane defined by $f(\boldsymbol{x})$ constant. This can be shown as follows: let C be any curve in the tangent space passing through $\boldsymbol{x}$, and let s be a parameter along C. Then, a unit tangent vector along C is given as: $\frac{\partial \boldsymbol{x}}{\partial s} = (\frac{\partial x_1}{\partial s}, \frac{\partial x_2}{\partial s}, \dots, \frac{\partial x_n}{\partial s})$. Further, we note that $\frac{df}{ds} = \frac{\partial f}{\partial \boldsymbol{x}}\frac{\partial \boldsymbol{x}}{\partial s} = \nabla f(\boldsymbol{x})^T \frac{\partial \boldsymbol{x}}{\partial s} = 0$, i.e., $\nabla f(\boldsymbol{x})$ is normal to $\frac{\partial \boldsymbol{x}}{\partial s}$..

The Hessian Matrix. The Hessian of f is a $n \times n$ matrix given by $\nabla^2 f(\boldsymbol{x})$,, where $[\nabla^2 f(\boldsymbol{x})]_{ij} = \frac{\partial^2 f}{\partial x_i \partial x_j}$. Note that Hessian is a symmetric matrix, since $\frac{\partial^2 f}{\partial x_i \partial x_j} = \frac{\partial^2 f}{\partial x_j \partial x_i}$..

As an example, we consider a quadratic function: $f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^T \boldsymbol{Q}\boldsymbol{x} - \boldsymbol{b}^T \boldsymbol{x}$ where $\boldsymbol{Q}$ is symmetric. Then its gradient and Hessian are given as: ; $\nabla f(\boldsymbol{x}) = \boldsymbol{Q}\boldsymbol{x}$; $\nabla^2 f(\boldsymbol{x}) = \boldsymbol{Q}$. .

Composite functions. Gradient and Hessian in the case of composite functions are computed as follows: Let $f(\boldsymbol{x}) = g(\boldsymbol{x})h(\boldsymbol{x})$ be a product of two functions; then,

$$\nabla f(\boldsymbol{x}) = \nabla g(\boldsymbol{x})h(\boldsymbol{x}) + g(\boldsymbol{x})\nabla h(\boldsymbol{x})$$
$$\nabla^2 f(\boldsymbol{x}) = \nabla^2 g(\boldsymbol{x})h(\boldsymbol{x}) + g(\boldsymbol{x})\nabla^2 h(\boldsymbol{x}) + \nabla g(\boldsymbol{x})\nabla h(\boldsymbol{x})^T + \nabla g(\boldsymbol{x})\nabla h(\boldsymbol{x})^T$$

Download free eBooks at bookboon.com

For vector-valued functions, let ∇f be a matrix defined by $[\nabla f(\boldsymbol{x})]_{ij} = \frac{\partial f_j(\boldsymbol{x})}{\partial x_i}$, then $\nabla f(\boldsymbol{x})^T$, then defines the Jacobian of f at point $\boldsymbol{x}$. Further, let $f = \boldsymbol{g}^T\boldsymbol{h}$, where $\boldsymbol{g}, \boldsymbol{h}$: $\mathbb{R}^n \to \mathbb{R}^m$, then:

$$\nabla f = [\nabla \boldsymbol{h}]\boldsymbol{g} + [\nabla \boldsymbol{g}]\boldsymbol{h}$$

Taylor series expansion for multi-variable functions. Taylor series expansion in the case of a multi-variable function is given as:

$$f(\boldsymbol{x}_0 + \boldsymbol{d}) = f(\boldsymbol{x}_0) + \nabla f(\boldsymbol{x}_0)^T\boldsymbol{d} + \frac{1}{2!}\boldsymbol{d}^T\nabla^2 f(\boldsymbol{x}_0)\boldsymbol{d} + \cdots$$

where $\nabla f(\boldsymbol{x}_0)$ and $\nabla^2 f(\boldsymbol{x}_0)$ are, respectively, the gradient and Hessian of f computed at $\boldsymbol{x}_0$. In particular, a first-order change in $f(\boldsymbol{x})$ at $\boldsymbol{x}_0$ along $\boldsymbol{d}$ is given as: $\delta f = \nabla f(\boldsymbol{x}_0)^T\boldsymbol{d}$,, where $\nabla f(\boldsymbol{x}_0)^T\boldsymbol{d}$ defines the directional derivative of $f(\boldsymbol{x})$ at $\boldsymbol{x}_0$ at along $\boldsymbol{d}$.

2.5 Convex Optimization Problems

Convex optimization problems are easier to solve due to the fact that convex functions have a unique global minimum. As defined in Sec. 2.2 above, a function $f(x)$ defined on a convex set is convex if and only if for all $\boldsymbol{x}, \boldsymbol{y} \in S,\ f(\alpha\boldsymbol{x} + (1-\alpha)\boldsymbol{y}) \leq \alpha f(\boldsymbol{x}) + (1-\alpha)f(\boldsymbol{y}), \alpha \in [0,1]$. In general, this condition may be hard to verify and other conditions based on properties of convex functions have been developed.

Convex functions have following important properties:

1. If $f \in \mathrm{C}^1$ (i.e., is differentiable), then f is convex over a convex set S if and only if for all $\boldsymbol{x}, \boldsymbol{y} \in S,\ f(\boldsymbol{y}) \geq f(\boldsymbol{x}) + \nabla f(\boldsymbol{x})^T(\boldsymbol{y} - \boldsymbol{x})$. Graphically, it means that a function is on or above the tangent hyperplane (line in two dimensions) passing through $\boldsymbol{x}$.
2. If $f \in \mathrm{C}^2$ (i.e., is twice differentiable), then f is convex over a convex set S if and only if for all $\boldsymbol{x} \in S,\ f''(\boldsymbol{x}) \geq 0$. In the case of multivariable functions, f is convex over a convex set S if and only if its Hessian matrix is positive semi-definite everywhere in S, i.e., for all $\boldsymbol{x} \in S$ and for all $\boldsymbol{d},\ \boldsymbol{d}^T\nabla^2 f(\boldsymbol{x})\boldsymbol{d} \geq 0$.
   This can be seen by considering second order Taylor series expansion of $f(\boldsymbol{x})$ at two points equidistant from a midpoint, $\overline{\boldsymbol{x}}$, given as: $f(\overline{\boldsymbol{x}} \pm \boldsymbol{d}) \cong f(\overline{\boldsymbol{x}}) \pm \nabla f(\overline{\boldsymbol{x}})^T\boldsymbol{d} + \frac{1}{2}\boldsymbol{d}^T\nabla^2 f(\overline{\boldsymbol{x}})\boldsymbol{d}$. Adding these two points with $\alpha = \frac{1}{2}$ and applying the definition of convex function gives: $f(\overline{\boldsymbol{x}}) \leq f(\overline{\boldsymbol{x}}) + \boldsymbol{d}^T\nabla^2 f(\overline{\boldsymbol{x}})\boldsymbol{d}$, or $\boldsymbol{d}^T\nabla^2 f(\overline{\boldsymbol{x}})\boldsymbol{d} \geq \boldsymbol{0}$.
3. If the Hessian is positive definite, i.e., for all $\boldsymbol{x} \in S$ and for all $\boldsymbol{d},\ \boldsymbol{d}^T\nabla^2 f(\boldsymbol{x})\boldsymbol{d} > 0,$ then the function is strictly convex. This is, however, a sufficient but not necessary condition, and a strictly convex function may have only a positive semidefinite Hessian at some points.

Download free eBooks at bookboon.com

4. If $f(\boldsymbol{x}^*)$ is a local minimum for a convex function *f* defined over a convex set *S*, then it is also a global minimum. This can be shown as follows: assume that $f(\boldsymbol{x}^*) = 0$, and replace $\boldsymbol{x}$ with $\boldsymbol{x}^*$ in property one above to get: $f(\boldsymbol{x}) \geq f(\boldsymbol{x}^*), \boldsymbol{x} \in S$. Thus, for a convex function ***f***, any point $\boldsymbol{x}^*$ that satisfies the necessary condition: $\nabla f(\boldsymbol{x}^*) = 0$, is a global minimum of *f*.

Due to the fact that convex functions have a unique global minimum, convexity plays an important role in optimization. For example, in numerical optimization convexity assures a global minimum to the problem. It is therefore important to first establish the convexity property when solving optimization problems. The following characterization of convexity applies to the solution spaces in such problems. Further ways of establishing convexity are discussed in (Boyd & Vandenberghe, Chaps. 2&3).

If a function $g_i(\boldsymbol{x})$ is convex, then the set $g_i(\boldsymbol{x}) \leq e_i$ is convex. Further, if functions $g_i(\boldsymbol{x}),\ i = 1, \dots, m,$ are convex, then the set $\{\boldsymbol{x}\colon g_i(\boldsymbol{x}) \leq e_i,\ i = 1, \dots, m\}$ is convex. In general, finite intersection of convex sets (that include hyperplanes and halfspaces) is convex.

For general optimization problems involving inequality constraints: $g_i(\boldsymbol{x}) \leq e_i,\ i = 1, \dots, m,$ and l, and equality constraints: $h_j(\boldsymbol{x}) = b_j,\ j = 1, \dots, l,$ the feasible region for the problem is defined by the set: $S = \{\boldsymbol{x}\colon g_i(\boldsymbol{x}) \leq e_i,\ h_j(\boldsymbol{x}) = b_j\}$. The feasible region is a convex set if the functions: $g_i\ i = 1, \dots, m,$ are convex and the functions: $h_j,\ j = 1, \dots, l,$ are linear. Note that these convexity conditions are sufficient but not necessary.

Download free eBooks at bookboon.com

2.6 Vector and Matrix Norms

Norms provide a measure for the size of a vector or matrix, similar to the notion of absolute value in the case of real numbers. A norm of a vector or matrix is a real-valued function with the following properties:

1. $\|\boldsymbol{x}\| \geq 0$ for all $\boldsymbol{x}$
2. $\|\boldsymbol{x}\| = 0$ if and only if $\boldsymbol{x} = \boldsymbol{0}$
3. $\|\alpha\boldsymbol{x}\| = |\alpha|\|\boldsymbol{x}\|$ for all $\alpha \in \mathbb{R}$
4. $\|\boldsymbol{x} + \boldsymbol{y}\| \leq \|\boldsymbol{x}\| + \|\boldsymbol{y}\|$

Matrix norms additionally satisfy:

5. $||\boldsymbol{AB}|| \leq ||\boldsymbol{A}||\ ||\boldsymbol{B}||$

Vector Norms. Vector p-norms are defined by $\|\boldsymbol{x}\|_p = (\sum_{i=1}^{n}|x_i|)^{\frac{1}{p}}, p \geq 1$. They include the 1-norm $\|\boldsymbol{x}\|_1 = \sum_{i=1}^{n}|x_i|$, the Euclidean norm $\|\boldsymbol{x}\|_2 = \sqrt{\sum_{i=1}^{n}|x_i|^2}$, and the ∞-norm $||\boldsymbol{x}||\infty = \max_{1\leq i\leq n}|x_i|$.

Matrix Norms. Popular matrix norms are induced from vector norms, given as: $||\boldsymbol{A}|| = \max_{||x||=1}||\boldsymbol{Ax}||$ All induced norms satisfy $||\boldsymbol{Ax}|| \leq ||\boldsymbol{A}||\ ||\boldsymbol{x}||$. Examples of induced matrix norms are:

1. $||\boldsymbol{A}||_1 = \max_{1\leq i<n} \Sigma_{i=1}^{n}|\boldsymbol{A}_{ij}|$ (the largest column sum of $\boldsymbol{A}$)
2. $||\boldsymbol{A}||_2 = \sqrt{\lambda_{max}(\boldsymbol{A}^T\boldsymbol{A})}$, where denotes the maximum eigenvalue of the matrix
3. $||\boldsymbol{A}||_\infty = \max_{1\leq i<n} \Sigma_{j=1}^{n}|\boldsymbol{A}_{ij}|$ (the largest row sum of $\boldsymbol{A}$)

2.7 Matrix Eigenvalues and Singular Values

Let $\boldsymbol{A}$ be an $n \times n$ matrix and assume that for some vector $\boldsymbol{v}$ and scalar λ, $\boldsymbol{Av} = \lambda\boldsymbol{v}$; then λ is an eigenvalue and $\boldsymbol{v}$ is an eigenvector of $\boldsymbol{A}$. The eigenvalues of $\boldsymbol{A}$ may be solved from:det $(\boldsymbol{A} - \lambda\boldsymbol{I}) = 0$. The nth degree polynomial on the left-hand side of the equation is the characteristic polynomial of $\boldsymbol{A}$ whose roots are the eigenvalues of $\boldsymbol{A}$. Let these roots be given as: $\lambda_i, i = 1,\ldots,n$ then their associated eigenvectors are solved from: $(\boldsymbol{A} - \lambda_i\boldsymbol{I})\ \boldsymbol{v} = 0$.

A matrix with repeated eigenvalues may not have a full set of eigenvectors which, by definition, are linearly independent. This happens, for instance, when the nullity of $(\boldsymbol{A} - \lambda_i\boldsymbol{I})$ is less than the degree of repetition of λ_i. In such cases, generalized eigenvectors may be substituted to make up the count.

Spectral Decomposition of a Symmetric Matrix. If $\boldsymbol{A}$ is symmetric, it has real eigenvalues and a full set of eigenvectors. Labeling them $\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_n$, it is possible to choose them to be orthonormal, such that $\boldsymbol{v}_i^T\boldsymbol{v}_i$=1, and $\boldsymbol{v}_i^T\boldsymbol{v}_j = 0$ for $i \neq j$. By defining $\boldsymbol{V} = (\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_n)$ and $\boldsymbol{\Lambda} = diag(\lambda_1, \lambda_2, \ldots, \lambda_n)$, we have $\boldsymbol{AV} = \boldsymbol{\Lambda V}$ or $\boldsymbol{A} = \boldsymbol{V\Lambda V}^T$. This is referred to as spectral decomposition of $\boldsymbol{A}$.

Download free eBooks at bookboon.com

Singular Value Decomposition of a Non-square Matrix. For non-square $\boldsymbol{A} \in \mathbb{R}^{m\times n}$, the singular value decomposition (SVD) of $\boldsymbol{A}$ is given as: $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T = \sum_{i=1}^{r} \sigma_i \boldsymbol{u}_i \boldsymbol{v}_i^T$, where $r = \text{rank}\ (\boldsymbol{A})$, $\boldsymbol{U} \in \mathbb{R}^{m\times r}$, $\boldsymbol{U}^T\boldsymbol{U} = \boldsymbol{I}_{m\times m}$; $\boldsymbol{V} \in \mathbb{R}^{n\times r}, \boldsymbol{V}^T\boldsymbol{V} = \boldsymbol{I}_{n\times n}$; $\boldsymbol{\Sigma} = \text{diag}\ (\sigma_1, \sigma_2, \dots, \sigma_r)$, where $\sigma_1 \geq \sigma_2 \geq \cdots, \geq \sigma_r$ are termed as singular values of $\boldsymbol{A}$.

2.8 Quadratic Function Forms

The function $f(\boldsymbol{x}) = \boldsymbol{x}^T\boldsymbol{Q}\boldsymbol{x} = \sum_{i=1}^{n}\sum_{j=1}^{n} Q_{i,j}x_i x_j$ describes a quadratic form. Quadratic forms in one and two variables are, respectively, given as: $f(x) = qx^2$ and $f(x_1, x_2) = Q_{1,1}x_1^2 + Q_{2,2}x_2^2 + 2Q_{1,2}x_1x_2$.

We note that replacing matrix $\boldsymbol{Q}$ by its symmetric counterpart $\frac{1}{2}(\boldsymbol{Q} + \boldsymbol{Q}^T)$ does not change $f(\boldsymbol{x})$. Therefore, in a quadratic form $\boldsymbol{Q}$ can always assumed to be symmetric.

The quadratic form is classified as:

a) Positive definite if $\boldsymbol{x}^T\boldsymbol{Q}\boldsymbol{x} > 0$
b) Positive semidefinite if $\boldsymbol{x}^T\boldsymbol{Q}\boldsymbol{x} \geq 0$
c) Negative definite if $\boldsymbol{x}^T\boldsymbol{Q}\boldsymbol{x} < 0$
d) Negative semidefinite if $\boldsymbol{x}^T\boldsymbol{Q}\boldsymbol{x} \leq 0$
e) Infinite otherwise

Let λ_{min} and λ_{max} and denote the minimum and maximum eigenvalues of $\boldsymbol{Q}$; then the quadratic form obeys:

$$\lambda_{min}\boldsymbol{x}^T\boldsymbol{x} \leq \boldsymbol{x}^T\boldsymbol{Q}\boldsymbol{x} \leq \lambda_{max}\boldsymbol{x}^T\boldsymbol{x}$$

Thus, positive definiteness of $\boldsymbol{x}^T\boldsymbol{Q}\boldsymbol{x}$ can be determined from the positivity of the eigenvalues of $\boldsymbol{Q}$. In particular, let $\lambda_i, i = 1,2, \dots, n$ be the eigenvalues of $\boldsymbol{Q}$; then $\boldsymbol{Q}$ is:

a) Positive definite only if $\lambda_i > 0, i = 1,2, \dots, n$
b) Positive semidefinite only if $\lambda_i \geq 0, i = 1,2, \dots, n$
c) Negative definite only if $\lambda_i < 0, i = 1,2, \dots, n$
d) Negative semidefinite only if $\lambda_i \leq 0, i = 1,2, \dots, n$
e) Indefinite otherwise

Geometrically, the set $S = \{\boldsymbol{x}: \boldsymbol{x}^T\boldsymbol{Q}\boldsymbol{x} \leq c\}$ describes an ellipsoid in $\mathbb{R}^n$ centered at $\boldsymbol{0}$ with its maximum eccentricity given by $\sqrt{\lambda_{max}/\lambda_{min}}$.

Download free eBooks at bookboon.com

2.9 Linear Systems of Equations

Systems of linear equations arise in solving the linear programming problems (Chapter 5). In the following, we briefly discuss the existence of solutions in the case of such systems.

Consider a system of m (independent) linear equations in n unknowns described as: $\boldsymbol{Ax} = \boldsymbol{b}$. Then, from linear algebra, the system has a unique solution if $m = n$; multiple solutions if $m < n$; and, the system is over-determined (and can be solved in the least-squares sense) if $m > n$.

For $m = n$, Gaussian elimination with partial pivoting results in a matrix decomposition $\boldsymbol{A} = \boldsymbol{PLU}$ where $\boldsymbol{P}, \boldsymbol{P}^T\boldsymbol{P} = \boldsymbol{I}$ is a permutation matrix; $\boldsymbol{L}$ is a lower triangular matrix with ones on the diagonal; and $\boldsymbol{U}$ is an upper triangular with eigenvalues of $\boldsymbol{A}$ on the main diagonal (Griva, Nash & Sofer, p.669). Then, using $\boldsymbol{y}, \boldsymbol{z}$ as intermediate variables, the system can be solved in steps as: $\boldsymbol{Pz} = \boldsymbol{b}$, $\boldsymbol{Ly} = \boldsymbol{z}$, $\boldsymbol{Ux} = \boldsymbol{y}$. If $\boldsymbol{A}$ is symmetric and positive definite, then Gaussian elimination results in $\boldsymbol{A} = \boldsymbol{LU} = \boldsymbol{LDL}^T$ where $\boldsymbol{D}$ is a diagonal matrix of (positive) eigenvalues of $\boldsymbol{A}$. In this case, the solution to the linear system is given as: $\boldsymbol{x} = \boldsymbol{LD}^{-1}\boldsymbol{L}^T\boldsymbol{b}$.

Download free eBooks at bookboon.com

Assume now that $m < n$; and that matrix $\boldsymbol{A}$ has full row rank. Then, we can arbitrarily choose $(n-m)$ variables as independent (nonbasic) variables, and solve the remaining (m) variables as dependent (basic) variables. The Gauss-Jordan elimination can be used to convert the system of equations into its canonical form given as: $\boldsymbol{I}_{(m)}\boldsymbol{x}_{(m)} + \boldsymbol{Q}\boldsymbol{x}_{(n-m)} = \boldsymbol{b}$. Then, the general solution to the linear system includes the independent variables: $\boldsymbol{x}_{(n-m)}$, and the dependent variables: $\boldsymbol{x}_{(m)} = \boldsymbol{b} - \boldsymbol{Q}\boldsymbol{x}_{(n-m)}$. A particular solution to the linear system can be obtained by setting: $\boldsymbol{x}_{(n-m)} = \boldsymbol{0}$, and obtaining: $\boldsymbol{x}_{(m)} = \boldsymbol{b}$.

For non-square matrices with $m > n$, Gram-Schmidt orthogonalization or Householder transformations can be applied to obtain $\boldsymbol{A} = \boldsymbol{Q}\boldsymbol{R}$, where $\boldsymbol{Q}\boldsymbol{Q}^T = \boldsymbol{I}$, and $\boldsymbol{R}$, where , and is upper triangular (QR factorization). The original system is equivalent to $\boldsymbol{R}\boldsymbol{x} = \boldsymbol{Q}^T\boldsymbol{b}$, which can then be solved via back-substitution. Following are two examples of practical situations that result in linear least-squares problems involving over-determined systems of linear equations $(m > n)$.

Linear Estimation Problem. Originally tackled by Carl Frederic Gauss, the linear estimation problem arises when estimating the state $\boldsymbol{x}$ of a linear system using a set of observations denoted as $\boldsymbol{y}$.

Consider a linear system of equations: $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{y}, \boldsymbol{A} \in \mathbb{R}^{m\times n}, m > n$ where $\boldsymbol{y}$ is the observation vector. Let $\boldsymbol{r} = \boldsymbol{A}\boldsymbol{x} - \boldsymbol{y}$ define a residual vector, and consider the unconstrained minimization problem: $\min_{\boldsymbol{x}} \|\boldsymbol{r}\|^2 = (\boldsymbol{A}\boldsymbol{x} - \boldsymbol{y})^T(\boldsymbol{A}\boldsymbol{x} - \boldsymbol{y})$.

Using derivatives, the problem is solved as: $\frac{d}{d\boldsymbol{x}}[\boldsymbol{x}^T\boldsymbol{A}^T\boldsymbol{A}\boldsymbol{x} - \boldsymbol{y}^T\boldsymbol{A}\boldsymbol{x} - \boldsymbol{x}^T\boldsymbol{A}\boldsymbol{y} + \boldsymbol{y}^T\boldsymbol{y}] = 0,$ which leads to: $\boldsymbol{A}^T\boldsymbol{A}\boldsymbol{x} = \boldsymbol{A}^T\boldsymbol{y}$. Thus, the solution to the least-squares problem is given as: $\hat{\boldsymbol{x}} = (\boldsymbol{A}^T\boldsymbol{A})^{-1}\boldsymbol{A}^T\boldsymbol{y},$, where had denotes the estimated value of the variable. Further, let $\boldsymbol{R}$ describe the measurement covariance matrix: $\boldsymbol{R} = E[\boldsymbol{r}\boldsymbol{r}^T]$. Then, the best linear estimator for $\boldsymbol{x}$ is given as: $\hat{\boldsymbol{x}} = (\boldsymbol{A}^T\boldsymbol{R}^{-1}\boldsymbol{A})^{-1}\boldsymbol{A}^T\boldsymbol{R}^{-1}\boldsymbol{b}$. .

Data Fitting Problem. The data-fitting problem involves fitting an nth degree polynomial given as: $p(x) = p_0 + p_1x + \cdots + p_nx^n$ to a set of data points: where $(x_i, y_i), i = 1, \ldots, N$ where $N > n$. .

To solve this problem, we similarly define a residual: $r_i = y_i - p(x_i) = y_i - (p_0 + p_1x_i + \cdots + p_nx_i^n)$, and define the following unconstrained minimization problem: $\min_{p_j}\sum_{i=1}^{N} r_i^2$, where p_j represents the coefficients of the polynomial. Then, by defining a coefficient vector: $\boldsymbol{x} = [p_0,\ p_1,\ \ldots,\ p_n]^T,$, and an $N \times (n+1)$ matrix $\boldsymbol{A}$ whose rows are observation vectors of the form $[1,\ x_i,\ x_i^2,\ \ldots,\ x_i^n],$, we can solve for the coefficients using the linear least-squares framework.

For example, in the linear case, $p(x) = p_0 + p_1x$, and $\boldsymbol{A}$ is a $N \times 2$ matrix whose rows are $[1, x_i]$ vectors. The least-squares method then results in the following equations:

$$\begin{pmatrix} \sum_{i=1}^{N} 1 & \sum_{i=1}^{N} x_i \\ \sum_{i=1}^{N} x_i & \sum_{i=1}^{N} x_i^2 \end{pmatrix}\begin{pmatrix} p_0 \\ p_1 \end{pmatrix} = \begin{pmatrix} \sum_{i=1}^{N} y_i \\ \sum_{i=1}^{N} x_iy_i \end{pmatrix}$$

Download free eBooks at bookboon.com

Using averages: $\frac{1}{N}\sum_{i=1}^{N} x_i = \bar{x},\ \frac{1}{N}\sum_{i=1}^{N} y_i = \bar{y},$ the solution is given as:

$$p_1 = \frac{\sum_{i=1}^{N}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{N}(x_i - \bar{x})^2};\quad p_0 = \bar{y} - p_1\bar{x}$$

Finally, the above solution can also be obtained through application of optimality conditions (Chapter 3).

2.10 Linear Diophantine System of Equations

A Linear Diophantine system of equations (LDSE) is represented as: $\boldsymbol{Ax} = \boldsymbol{b},\ \boldsymbol{x} \in \mathbb{Z}^n$. The following algebra concepts are needed to formulate and solve problems involving a solution to LDSE.

Unimodular Matrices. Matrix $\boldsymbol{A} \in \mathbb{Z}^{n\times n}$ is unimodular if $\det(\boldsymbol{A}) = \pm 1$. Further, if $\boldsymbol{A} \in \mathbb{Z}^{n\times n}$ is unimodular, then $\boldsymbol{A}^{-1} \in \mathbb{Z}^{n\times n}$. Matrix $\boldsymbol{A} \in \mathbb{Z}^{n\times n}$ is totally unimodular if every square submatrix $\boldsymbol{C}$ of $\boldsymbol{A}$, has $\det(\boldsymbol{C}) \in \{0, \pm 1\}$.

Hermite Normal Form of a Matrix. Let $\boldsymbol{A} \in \mathbb{Z}^{m\times n}, rank(\boldsymbol{A}) = m;$ then, $\boldsymbol{A}$ has a unique hermite normal form given as: $\mathrm{HNF}(\boldsymbol{A}) = [\boldsymbol{D}\ \boldsymbol{0}]$, where $\boldsymbol{D}$ is lower triangular with $d_{ij} < d_{ii}, j < i.$. Further, there exists a unimodular matrix $\boldsymbol{U}$ such that $\boldsymbol{AU} = \mathrm{HNF}(\boldsymbol{A})$, where we note that post-multiplication by a unimodular matrix involves performing elementary column operations. Moreover, let $\boldsymbol{u}_1, \boldsymbol{u}_2, \ldots, \boldsymbol{u}_n$ represent the $\boldsymbol{U}$ columns of then $\{\boldsymbol{u}_{m+1}, \ldots, \boldsymbol{u}_n\}$ form a basis for ker $(\boldsymbol{A})$.

Solution to the LDSE. Assume that $\boldsymbol{A} \in \mathbb{Z}^{m\times n}, rank(\boldsymbol{A}) = m,$, and let $\boldsymbol{AU} = \mathrm{HNF}(\boldsymbol{A})$; then, we may consider: $\boldsymbol{b} = \boldsymbol{Ax} = \boldsymbol{AUU}^{-1}\boldsymbol{x} = \boldsymbol{HNF}(\boldsymbol{A})\boldsymbol{y},\ \boldsymbol{y} = \boldsymbol{U}^{-1}\boldsymbol{x}$. Assume that we have a solution $\boldsymbol{y}_0$ to: $\mathrm{HNF}(\boldsymbol{A})\boldsymbol{y}_0 = \boldsymbol{b}$; then, the general solution to the LDSE is given as: $\boldsymbol{x} = \boldsymbol{x}_0 + \sum_{i=1}^{n-m} \alpha_i \boldsymbol{x}_i$, where $\boldsymbol{x}_0 = \boldsymbol{U}\boldsymbol{y}_0, \boldsymbol{x}_i \in \{\boldsymbol{u}_{m+1}, \ldots, \boldsymbol{u}_n\}$.

2.11 Condition Number and Convergence Rates

The condition number of a matrix is defined as: $cond\ (\boldsymbol{A}) = \|\boldsymbol{A}\| \cdot \|\boldsymbol{A}^{-1}\|$. Note that and $cond\ (\boldsymbol{A}) \geq 1$ and $cond\ (\boldsymbol{I}) = 1$, where $\boldsymbol{I}$ is an identity matrix. If $\boldsymbol{A}$ is symmetric with real eigenvalues, and 2-norm is used, then $(\boldsymbol{A}) = \lambda_{max}(\boldsymbol{A})/\lambda_{min}(\boldsymbol{A})$.

The condition number of the Hessian matrix affects the convergence rates of the optimization algorithms. Ill-conditioned matrices give rise to numerical errors in computations. In certain cases, it is possible to improve the condition number by scaling the variables. The convergence property implies that the generated sequence converges to the true solution in the limit. The rate of convergence dictates how quickly the approximate solutions approach the exact solution.

Download free eBooks at bookboon.com

Assume that a sequence of points $\{x^k\}$ converges to a solution point x^* and define an error sequence: $e_k = x^k - x^*$. Then, we say that the sequence $\{x^k\}$ converges to x^* with rate and rate constant C if $\lim_{k\to\infty} \frac{\|e_{k+1}\|}{\|e_k\|^r} = C$. Further, if uniform convergence is assumed, then $\|e_{k+1}\| = C\|e_k\|^r$ holds for all k. Thus, convergence to the limit point is faster if r is larger and C is smaller. Specific cases for different choices of r and C are mentioned below.

Linear convergence. For $r = 1$ and $0 < C < 1$, $\|e_{k+1}\| = C\|e_k\|$, signifying linear convergence. In this case the speed of convergence depends only on C, which can be estimated as $C \approx \frac{f(x^{k+1})-f(x^*)}{f(x^k)-f(x^*)}$. .

Quadratic Convergence. For r = 2, the convergence is quadratic, i.e., $\|e_{k+1}\| = C\|e_k\|^2$. In this case if additionally C = 1, then the number of correct digits in double at every iteration.

Superlinear Convergence. For $1 < r < 2$, the convergence is superlinear. Superlinear convergence is achieved by numerical algorithms that only use the gradient (first derivative) of the cost function, and thus can qualitatively match quadratic convergence.

Download free eBooks at bookboon.com

2.12 Conjugate-Gradient Method for Linear Equations

The conjugate-gradient method is an iterative method designed to solve a system of linear equations described as: $\boldsymbol{Ax} = \boldsymbol{b}$, where $\boldsymbol{A}$ is assumed normal, i.e., $\boldsymbol{A}^T\boldsymbol{A} = \boldsymbol{A}\boldsymbol{A}^T$. The method initializes with $\boldsymbol{x}_0 = \boldsymbol{0}$, and uses an iterative process to obtain an approximate solution $\boldsymbol{x}_n$ in n iterations. The solution is exact in the case of quadratic functions of the form: $q(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}^T\boldsymbol{x}$. For general nonlinear functions, convergence in $2n$ iterations is to be expected. The method is named so because $\boldsymbol{Ax} - \boldsymbol{b}$ represents the gradient of the quadratic function. Solving a linear system of equations thus amounts to solving the minimization problem involving a quadratic function.

The conjugate-gradient method generates a set of vectors $\boldsymbol{v}_1, \boldsymbol{v}_2, \dots, \boldsymbol{v}_n$ that are conjugate with respect to $\boldsymbol{A}$ matrix, i.e., $\boldsymbol{v}_i^T\boldsymbol{A}\boldsymbol{v}_j = 0, i \neq j$. Let $\boldsymbol{v}_{-1} = \boldsymbol{0}$, $\beta_0 = 0$; and define a residual $\boldsymbol{r}_i = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_i$. Then, a set of conjugate vectors is iteratively generated as:

$$\boldsymbol{v}_i = \boldsymbol{r}_i + \beta_i\boldsymbol{v}_{i-1}, \quad \beta_i = \frac{\boldsymbol{v}_i^T\boldsymbol{A}\boldsymbol{r}_i}{\boldsymbol{v}_i^T\boldsymbol{A}\boldsymbol{v}_i}$$

We note that the set of conjugate vectors of a matrix is not unique. Further, nonzero conjugate vectors with respect to a positive-definite matrix are linearly independent.

In conjugate-gradient and other iterative methods, scaling of variables, termed as preconditioning, helps reduce the condition number of the coefficient matrix, which aids in fast convergence of the algorithm.

Towards that end, we consider a linear system of equations: $\boldsymbol{Ax} = \boldsymbol{b}$, and use a linear transformation to formulate an equivalent system that is easier to solve. Let $\boldsymbol{P}$ be any nonsingular $n \times n$ matrix, then an equivalent left-preconditioned system is formulated as: $\boldsymbol{P}^{-1}\boldsymbol{Ax} = \boldsymbol{P}^{-1}\boldsymbol{b}$, , and a right-preconditioned system is given as: $\boldsymbol{A}\boldsymbol{P}^{-1}\boldsymbol{P}\boldsymbol{x} = \boldsymbol{b}$. . As the operator $\boldsymbol{P}^{-1}$ is applied at each step of the iterative solution, it helps to choose a simple $\boldsymbol{P}^{-1}$ with a small computational cost. An example of a simple preconditioner is the Jacobi preconditioner: $\boldsymbol{P} = diag(\boldsymbol{A})$.

Further, if $\boldsymbol{A}$ is symmetric and positive-definite, then $\boldsymbol{P}^{-1}$ should be chosen likewise. If both $\boldsymbol{P}^{-1}$ and $\boldsymbol{A}$ are positive-definite, then we can use the Cholesky decomposition of $\boldsymbol{P}$, $\boldsymbol{P} = \boldsymbol{C}^T\boldsymbol{C}$,, to write $\boldsymbol{C}^{-1}\boldsymbol{C}^{-T}\boldsymbol{Ax} = \boldsymbol{C}^{-1}\boldsymbol{C}^{-T}\boldsymbol{b}$, or $\boldsymbol{C}^{-T}\boldsymbol{A}\boldsymbol{C}^{-1}\boldsymbol{x} = \boldsymbol{C}^{-T}\boldsymbol{b}$. Then, by defining $\boldsymbol{C}^{-T}\boldsymbol{A}\boldsymbol{C}^{-1} = \widehat{\boldsymbol{A}}$, $\boldsymbol{C}^{-T}\boldsymbol{b} = \widehat{\boldsymbol{b}}$, we obtain $\widehat{\boldsymbol{A}}\boldsymbol{x} = \widehat{\boldsymbol{b}}$, where $\widehat{\boldsymbol{A}}$ is positive-definite.

Download free eBooks at bookboon.com

2.13 Newton's Method for Nonlinear Equations

Newton's method, also known as Newton-Raphson method, iteratively solves a nonlinear equation: $f(x) = 0$, starting from an initial point x_0. The method generates a series of solutions $\{x_k\}$ that are expected to converge to a fixed point x^* that represents a root of the equation. To develop the method, we assume that an estimate of the solution is available as x_k, and use first order Taylor series to approximate $f(x)$ around x_k, i.e., let

$$f(x_k + \delta x) = f(x_k) + f'(x_k)\delta x$$

Then, by setting $f(x_k + \delta x) = 0$, we can solve for the offset δx, and use it to update our estimate x_k as:

$$x_{k+1} = x_k - f(x_k)/f'(x_k)$$

Next, Newton's method can be extended to a system of nonlinear equations, given as:

$$f_1(x_1, x_2, \dots, x_n) = 0$$

$$f_2(x_1, x_2, \dots, x_n) = 0$$
$$\vdots$$
$$f_n(x_1, x_2, \dots, x_n) = 0$$

Let a gradient matrix $\nabla f(\boldsymbol{x})$ be formed with columns: $\nabla f_1(\boldsymbol{x}), \nabla f_2(\boldsymbol{x}), \dots, \nabla f_n(\boldsymbol{x})$; then, the transpose of the gradient matrix defines the Jacobian matrix given as: $J(\boldsymbol{x}) = \nabla f(\boldsymbol{x})^T$. Using the Jacobian matrix, the update rule in the n-dimensional case is given as:

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k - \big(J(\boldsymbol{x}_k)\big)^{-1} f(\boldsymbol{x}_k).$$

Convergence Rate. We first note that Newton's method requires a good initial guess for it to converge. Newton's method, if it converges, exhibits quadratic rate of convergence near the solution point. The method can become unstable if $f(x^*) \approx 0$. Assuming $f'(x^*) \neq 0$, and x_k is sufficiently close to x^*, we can use second order Taylor series to write:

$$x_{k+1} - x^* \approx \frac{1}{2}\left(\frac{f''(x^*)}{f'(x^*)}\right)(x_k - x^*)^2$$

which shows that Newton's method has quadratic convergence with a rate constant: $C = \frac{1}{2}\left|\frac{f''(x^*)}{f'(x^*)}\right|$.

Download free eBooks at bookboon.com

3 Graphical Optimization

We briefly discuss the graphical optimization concepts in this chapter before proceeding to formal mathematical optimization method in Chapter 4 and computational methods in Chapter 7. Graphical approach is recommended for problems of low dimensions, typically those involving one or two variables. Apart from being simple, the graphical method provides a valuable insight into the problem, which may not be forthcoming in the case of mathematical and computational optimization methods, particularly in the case of two-dimensional problems.

The graphical method is applicable when the optimization problem is formulated with one or two variables. Graphical optimization helps enhance our understanding of the underlying problem and develop an appeal for the expected solution. The method involves plotting contours of the cost function over a feasible region enclosed by the constraint boundaries. In most cases, the desired optimum can be spotted by inspection.

Download free eBooks at bookboon.com

Software implementation of the graphical method uses a grid of paired values for the optimization variables to plot the objective function contours and the constraint boundaries. The minimum of the cost function can then be identified on the plot. Graphical minimization procedure thus involves the following steps:

1. Establishing the feasible region. This is done by plotting the constraint boundaries.
2. Plotting the level curves (or contours) of the cost function and identifying the minimum.

The graphical method is normally implemented in a computational software package such as Matlab © and Mathematica ©. Both packages include functions that aid the plotting and visualization of cost function contours and constraint boundaries. Code for Matlab implementation of graphical optimization examples considered in this chapter is provided in the Appedix.

Learning Objectives: The learning goals in this chapter are:

1. Recognize the usefulness and applicability of the graphical method.
2. Learn how to apply graphical optimization techniques to problems of low dimensions.

3.1 Functional Minimization in One-Dimension

Graphical function minimization in one-dimension is performed by computing and plotting the function values at a set of discrete points and identifying its minimum value on the plot. We assume that the feasible region for the problem is a closed interval: $S = [x_l, x_u]$; then, the procedure can be summarized as follows:

1. Define a grid over the feasible region: let $x = x_l + k\delta, k = 0,1,2, \ldots$ where δ defines the granularity of the grid.
2. Compute and compare the function values over the grid points to find the minimum.

An illustrative example for one-dimensional minimization is provided below.

Example 3.1: Graphical function minimization in one-dimension

Let the problem be defined as: Minimize e^x subject to $x^2 \leq 1$. Then, to find a solution, we define a grid over the feasible region as follows: let $= 0.01, x = -1, -0.99, \ldots, -0.01, 0, 0.01, \ldots, 0.99, 1$. Then, $f(x) = e^{-1}, e^{-0.99}, \ldots, e^{-0.01}, 1, e^{0.01}, \ldots, e^{0.99}, e^1$. By comparison, $f_{min} = e^{-1}$ at $x = -1$.

Download free eBooks at bookboon.com

3.2 Graphical Optimization in Two-Dimensions

Graphical optimization is most useful for optimization problems involving functions of two variables. Graphical function minimization in two-dimensions is performed by plotting the contours of the objective function along with the constraint boundaries on a two-dimensional grid. In Matlab ©, the grid points can be generated with the help of 'meshgrid' function. Mathematica © also provide similar capabilities.

In the following we discuss three examples of applying graphical method in engineering design optimization problems where each problem contains two optimization variables.

Example 3.2: Hollow cylindrical cantilever beam design (Arora, p. 85)

We consider the minimum-weight design of a cantilever beam of length *L,* with hollow circular cross-section (outer radius R_o, inner radius R_i) subjected to a point load *P.* The maximum bending moment on the beam is given as *PL,* the maximum bending stress is given as: $\sigma_a = \frac{PLR_o}{I}$, and the maximum shear stress is given as: $\tau = \frac{P}{3I}\left(R_o^2 + R_0R_i + R_i^2\right)$, where $I = \frac{\pi}{4}(R_o^4 - R_i^4)$ is the moment of inertia of the cross-section. The maximum allowable bending and shear stresses are given as σ_a and τ_a, respectively.

Let the design variables be selected as the outer radius R_o and the inner radius R_i; then, the optimization problem is stated as follows:

Minimize $f(R_0, R_i) = \pi\rho L(R_0^2 - R_i^2)$
Subject to: $\frac{\sigma}{\sigma_a} - 1 \leq 0,\ \frac{\tau}{\tau_a} - 1 \leq 0;\ \ R_0, R_i \leq 0.2m$

The following data are provided for the problem: $P = 10kN,\ L = 5m,\ \sigma_a = 250MPa,\ \tau_a = 90MPa,$ $E = 210GPa,\ \rho = 7850\ kg/m^3$. After substituting the values, and dropping the constant terms in *f,* the optimization problem is stated as:

Minimize $f(R_0, R_i) = R_0^2 - R_i^2$
Subject to: $g1$: $\frac{8\times10^{-4}R_o}{\pi(R_0^4 - R_i^4)} - 1 \leq 0;\ \ g2$: $\frac{4\left(R_o^2 + R_0R_i + R_i^2\right)}{27\pi\left(R_0^4 - R_i^4\right)} - 1 \leq 0;\ \ R_0, R_i \leq 20cm$

The graphical solution to the problem, obtained from Matlab, is shown in Figure 3.1. The optimal solution is given as: $R_o = 0.12m,\ R_i = 0.115m,\ f^* = 0.001175$.

Download free eBooks at bookboon.com

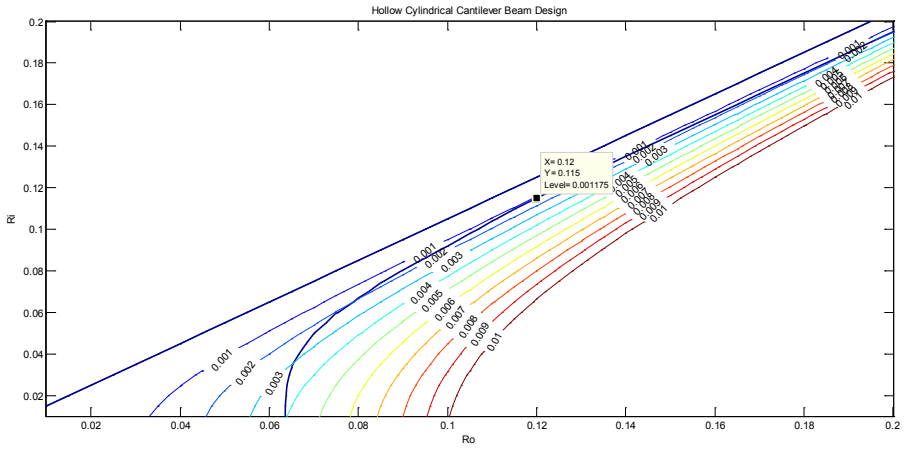


Figure 3.1: Graphical solution to the minimum`-weight hollow cantilever beam design (Example 3.2)

Example 3.3: Symmetrical two-bar truss design (Arora, p. 59)

We wish to design a symmetrical two-bar truss to withstand a load $W = 50kN$. The truss consists of two steel tubes pinned together at the top and supported on the ground at the other (figure). The truss has a fixed span $s = 2m$, and a height $h = \sqrt{l^2 - 1}$, where l is the length of the tubes; both tubes have a cross-sectional area: $A = 2\pi Rt$, where R is the radius of the tube and t is the thickness. The objective is to design a minimum-weight structure, where total weight is $2\rho lA$.

Download free eBooks at bookboon.com

The truss design is subject to the following constraints:

1. The height of the truss is to be limited as: $2 \leq h \leq 5$;
2. The tube thickness is to be limited as: $R \leq 45t$;
3. The maximum allowable stress is given as: $\sigma_a = 250MPa$;
4. To prevent buckling, tube loading should not exceed a critical value: $\frac{Wl}{2h} \leq \frac{P_{cr}}{FS} = \frac{1}{FS}\frac{\pi^2 EI}{(Kl)^2}$, where $K = 0.7,\ E = 210GPa$, the moment of inertia: $I \cong \pi R^3 t$, and $FS = 2$ denotes a safety factor.

Let the design variables be selected as: h, R, t; then, the optimization problem is formulated as:

Minimize $f(h,R,t) = 4\pi\rho\sqrt{h^2+1}Rt$
Subject to: $g1$: $\frac{W\sqrt{h^2+1}}{4\pi hRt\sigma_a} - 1 \leq 0,\ g2$: $\frac{0.49W(h^2+1)^{\frac{3}{2}}}{\pi^3 EhR^3 t} - 1 \leq 0,\ g3: R - 45t \leq 0,\ g4: 2 \leq h \leq 5.$

In the above formulation, there are three design variables: h, R, t. Consequently, we need to fix the value of one variable in order to perform the graphical design with two variables. We arbitrarily fix $h = 3m$, and graphically solve the resulting minimization problem stated, after dropping the constant terms in f, as follows:

Minimize $f(R,t) = Rt$
Subject to: $g1$: $\frac{1.677\times10^{-5}}{Rt} - 1 \leq 0,\ g2$: $\frac{3.966\times10^{-8}}{R^3 t} - 1 \leq 0,\ g3: R - 45t \leq 0$

A graph of the objective function and the constraints for the problem is shown in the Figure 3.2. From the figure, the optimum values of the design variables are: $R = 3.7cm,\ t = 0.8mm,\ f^* = 3 \times 10^{-5}$..

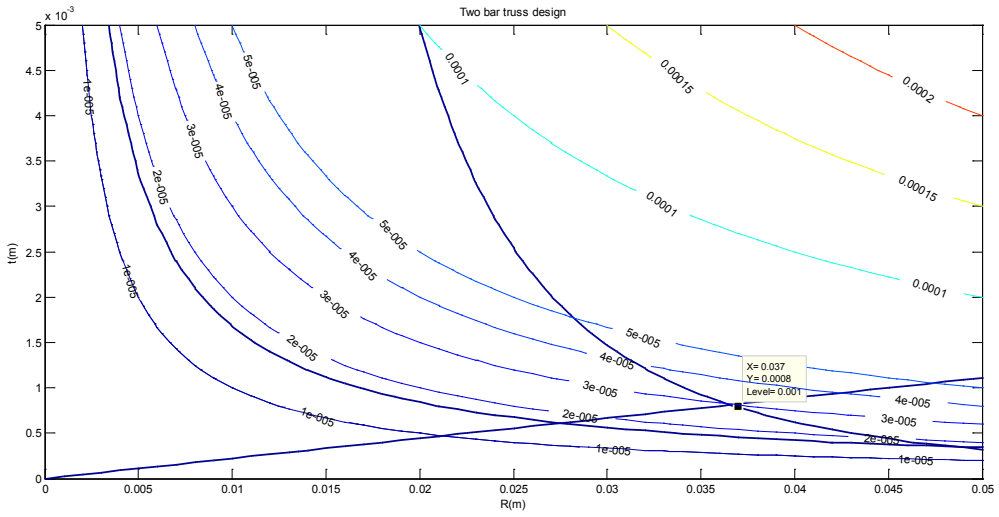


Figure 3.2: Graphical solution to the minimum-weight symmetrical two-bar truss design (Example 3.3)

Download free eBooks at bookboon.com

Example 3.4: Symmetrical three-bar truss design (Arora, p. 46, 86)

We consider the minimum-weight design of a symmetric three-bar truss supported over-head. Members 1 and 3 have the same cross-sectional area A_1 and the middle member 2 has cross-sectional area A_2. Let l be the height of the truss, then the lengths of member 1 and 3 are $\sqrt{2}l$ and that of member 2 is l.

A load P at the joint is applied at an angle θ, so that the horizontal and vertical components of the applied load are given, respectively, as: $P_u = P\cos\theta$, $P_v = P\sin\theta$. The design variables for the problem are selected as A_1 and A_2. The design objective is to minimize the total $mass = \rho l(2\sqrt{2}A_1 + A_2)$.

The constraints in the problem are formulated as follows:

a) The stresses in members 1, 2 and 3, computed as: $\sigma_1 = \frac{1}{\sqrt{2}}\left[\frac{P_u}{A_1} + \frac{P_v}{(A_1+\sqrt{2}A_2)}\right]$; $\sigma_2 = \frac{\sqrt{2}P_v}{(A_1+\sqrt{2}A_2)}$; $\sigma_3 = \frac{1}{\sqrt{2}}\left[-\frac{P_u}{A_1} + \frac{P_v}{(A_1+\sqrt{2}A_2)}\right]$, are to be limited by the allowable stress for the material.

b) The axial force in members under compression, given as: $F_i = \sigma_i A_i$, is limited by the buckling load, i.e., $-F_i \le \frac{\pi^2 EI}{l_i^2}$, or $-\sigma_i \le \frac{\pi^2 E\beta A_i}{l_i^2} \le \sigma_a$, or where the moment of inertia is estimated as: $I_i = \beta A_i^2$, $\beta =$ constant.

Download free eBooks at bookboon.com

c) The horizontal and vertical deflections of the load point, given as:
$u = \frac{\sqrt{2}lP_u}{A_1E},\ v = \frac{\sqrt{2}lP_v}{(A_1+\sqrt{2}A_2)E}$, are to be limited by $u \le \Delta_u,\ v \le \Delta_v$.

d) To avoid possible resonance, the lowest eigenvalue of the structure, given as:
$\zeta = \frac{3EA_1}{\rho l^2(4A_1+\sqrt{2}A_2)}$, where ρ is the mass density should be higher than a specified frequency, i.e., $\zeta \ge (2\pi\omega_0)^2$.

e) The design variables are required to be greater than some minimum value, i.e., $A_1, A_2 \ge A_{min}$.

For a particular problem, let $l = 1.0m,\ P = 100kN,\ \theta = 30°,\ \rho = 2800\frac{kg}{m^3},\ E = 70GPa,\ \sigma_a = 140MPa,\ \Delta_u = \Delta_v = 0.5\ cm,\ \omega_0 = 50Hz,\ \beta = 1.0,$ and $A_{min} = 2cm^2$. Then, $P_u = \frac{\sqrt{3}P}{2},\ P_v = \frac{P}{2}$; a and Then, and the resulting optimal design problem is formulated as:

Minimize $f(A_1, A_2) = 2\sqrt{2}A_1 + A_2$
Subject to:

$$g1: 2.5\times10^{-4}\left[\frac{\sqrt{3}}{A_1} + \frac{1}{(A_1+\sqrt{2}A_2)}\right] - 1 \le 0,$$
$$g2: 2.5\times10^{-4}\left[-\frac{\sqrt{3}}{A_1} + \frac{1}{(A_1+\sqrt{2}A_2)}\right] - 1 \le 0,$$
$$g3: \frac{5\times10^{-4}}{(A_1+\sqrt{2}A_2)} - 1 \le 0,$$
$$g4: 1.02\times10^{-7}\left[\frac{\sqrt{3}}{A_1^2} - \frac{1}{A_1(A_1+\sqrt{2}A_2)}\right] - 1 \le 0,$$
$$g5: \frac{3.5\times10^{-4}}{A_1} - 1 \le 0,$$
$$g6: \frac{2\times10^{-4}}{A_1+\sqrt{2}A_2} - 1 \le 0,$$
$$g7: \frac{2\times10^{-4}}{A_1} - 1 \le 0,$$
$$g8: \frac{2\times10^{-4}}{A_2} - 1 \le 0,$$
$$g9: 1.316\times10^{-5}(4A_1+\sqrt{2}A_2) - 1 \le 0,$$
$$g10: 2467A_1 - 1 \le 0.$$

The problem was graphically solved in Matlab (see Figure 3.3). The optimum solution is given as: $A_1 = A_3 = 6cm^2,\ A_2 = 2\ cm^2,\ f^* = 0.00486cm^2$.

Download free eBooks at bookboon.com

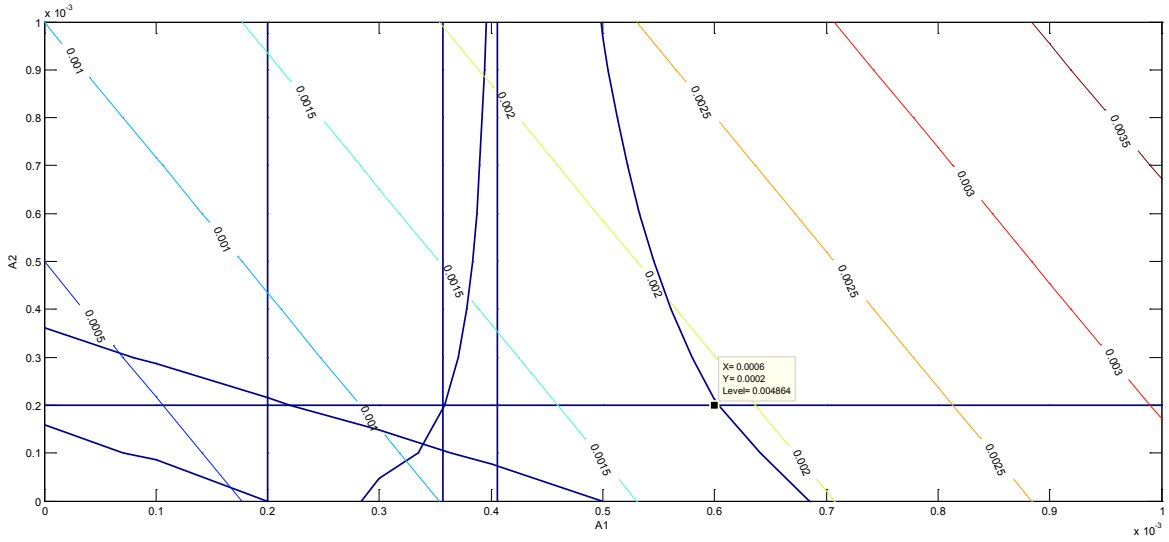


Figure 3.3: Graphical solution to the minimum-weight symmetrical three-bar truss design (Example 3.4)

Appendix to Chapter 3: Matlab Code for Examples 3.2–3.4

Example 3.2: Cantilever beam design (Arora, Prob. 2.23, p. 64)

```
% cantilever beam design, Prob. 2.23 (Arora)
ro=.01:.005:.2;
ri=.01:.005:.2;
[Ro,Ri]=meshgrid(ro,ri);
F=Ro.*Ro-Ri.*Ri;
G1=8e-4/pi*Ro./(Ro.^4-Ri.^4)-1;
G2=4/27/pi*(Ro.*Ro+Ro.*Ri+Ri.*Ri)./(Ro.^4-Ri.^4)-1;
figure, hold
contour(ro,ri,G1,[0 0])
contour(ro,ri,G2,[0 0])
[c,h]=contour(ro,ri, F, .001:.001:.01);
clabel(c,h);
```

Example 3.3: Two-bar truss design (Arora, Prob. 2.16, p. 61)

```
% two-bar trus; prob. 2.16 (Arora)
W=50e3;
r=0:.001:.05;
t=0:.0001:.005;
[R,T]=meshgrid(r,t);
F=R.*T;
G1=sqrt(10)*W/12/pi./(250e6*R.*T)-1;
```


Download free eBooks at bookboon.com

```
G2=4.9*sqrt(10)*W/3/pi/pi/pi./(210e9*R.^3.*T)-1;
G3=R-45*T;
figure, hold
contour(r,t,G1,[0 0]), pause
contour(r,t,G2,[0 0]), pause
contour(r,t,G3,[0 0]), pause
[c,h]=contour(r,t,F);
clabel(c,h)
```

Example 3.4: Symmetric three-bar truss (Arora, Prob. 3.29, p. 86)

```
%three-bar truss prob. 3.29 (Arora)
a1=0:1e-4:1e-3;
a2=0:1e-4:1e-3;
[A1,A2]=meshgrid(a1,a2);
F=2*sqrt(2)*A1+A2;
G1=2.5e-4*(sqrt(3)./A1+1./(A1+sqrt(2)*A2))-1;
G2=2.5e-4*(-sqrt(3)./A1+1./(A1+sqrt(2)*A2))-1;
G3=5e-4./(A1+sqrt(2)*A2)-1;
G4=1.02e-7*(sqrt(3)./A1./A1-1./A1./(A1+sqrt(2)*A2))-1;
G5=3.5e-4./A1-1;
G6=2e-4./(A1+sqrt(2)*A2)-1;
G7=2e-4./A1-1;
G8=2e-4./A2-1;
G9=1.316e-5*(A1+sqrt(2)*A2)-1;
G10=2467*A1-1;
figure, hold
contour(a1,a2, G1,[0 0]), pause
contour(a1,a2, G2,[0 0]), pause
contour(a1,a2, G3,[0 0]), pause
contour(a1,a2, G4,[0 0]), pause
contour(a1,a2, G5,[0 0]), pause
contour(a1,a2, G6,[0 0]), pause
contour(a1,a2, G7,[0 0]), pause
contour(a1,a2, G8,[0 0]), pause
contour(a1,a2, G9,[0 0]), pause
contour(a1,a2, G10,[0 0]), pause
[c,h]=contour(a1,a2,F);
clabel(c,h)
```


Download free eBooks at bookboon.com

4 Mathematical Optimization

In this chapter we discuss the mathematical optimization problem, including its formulation and the techniques to solve it. The mathematical optimization problem involves minimization (or maximization) of a real-valued cost function by systematically choosing the values of a set of variables that are subject to inequality and/or equality constraints. Both cost and constraint functions are assumed analytical so that they can be locally approximated by Taylor series and their first and second derivatives can be computed. The analytical techniques used to solve the optimization problem include determination of first and second order necessary conditions that reveal a set of possible candidate points, which are then evaluated using sufficient conditions for an optimum. In convex optimization problems the feasible region, i.e., the set of points that satisfy the constraints, is a convex set and both object and constraint functions are also convex. In such problems, the existence of a single global minimum is assured.

Learning Objectives: the learning goals in this chapter are:

1. Understand formulation of unconstrained and constrained optimization problems
2. Learn the application of first and second order necessary conditions to solve optimization problems
3. Learn solution techniques used for convex optimization problems

Download free eBooks at bookboon.com

4. Understand the geometric viewpoint associated with optimization algorithms
5. Understand the concept of Lagrangian duality and how it helps toward finding a solution.
6. Learn the techniques used for post-optimality analysis for nonlinear problems

4.1 The Optimization Problem

The general nonlinear optimization problem (the nonlinear programming problem) is defined as:

$$\min_{\boldsymbol{x}} f(\boldsymbol{x})$$
$$\text{Subject to} \begin{cases} h_i(\boldsymbol{x}) = 0, \ i = 1, \dots, l; \\ g_j(\boldsymbol{x}) \leq 0, j = i, \dots, m; \\ x_{iL} \leq x_i \leq x_{iU}, \ i = 1, \dots, n \end{cases} \tag{4.1}$$

The above problem assumes minimization of a multi-variable scalar cost function $f(\boldsymbol{x})$, where $\boldsymbol{x} \in \mathbb{R}^n$, $\boldsymbol{x}^T = [x_1, x_2, \dots, x_n]$, that is subjected to l equality constraints and m_i inequality constraints. Additionally, lower and upper bounds on the optimization variables are considered, where these bounds may be grouped with the inequality constraints.

Special cases involving variants of the general problem can also be considered. For example, the absence of both equality and inequality constraints specifies an unconstrained optimization problem; the problem may only involve a single type of constraints; the linearity of the objective and constraint functions specifies a linear programming problem (discussed in Chapter 5); and the restriction of optimization variables to a discrete set of values specifies a discrete optimization problem (discussed in Chapter 6).

We begin with defining the feasible region for the optimization problem and a discussion of the existence of points of minima or maxima of the objective function in that region.

Feasible Region. The set $\Omega = \{x: h_i(\boldsymbol{x}) = 0, \ g_j(\boldsymbol{x}) \leq 0, \ x_{iL} \leq x_i \leq x_{iU}\}$ is termed as the feasible region for the problem. If the feasible region is convex, and additionally h_i, g_j are convex functions, then the problem is a convex optimization problem with some obvious advantages, e.g., f only has a single global minimum in Ω.

The Extreme Value Theorem in Calculus (attributed to Karl Weierstrass) provides sufficient conditions for the existence of minimum (or maximum) of a function defined over a complex domain. The theorem states: *A continuous function $f(\boldsymbol{x})$ defined over a closed and bounded set $\Omega \subseteq D(f)$ attains its maximum and minimum in Ω.*

Thus, according to this theorem, if the feasible region Ω of the problem is closed and bounded, a minimum for the problem exists. The rest of the book discusses various ways to find that minimum.

Download free eBooks at bookboon.com

Finding the minimum is relatively easy in the case of linear programming problems, but could be considerably difficult in the case of nonlinear problems with an irregular of the constraint surface. As a consequence, numerical methods applied to a nonlinear problem may only return a local minimum. Stochastic methods, such as Simulated Annealing, have been developed to find a global minimum with some certainty in the case of nonlinear problems. These methods are, however, not covered in this text.

Finally, we note that the convexity property, if present, helps in finding a solution to the optimization problem. If convexity can be ascertained through application of appropriate techniques, then we are at least assured that any solution found in the process would be the global solution.

4.2 Optimality criteria for the Unconstrained Problems

We begin by reviewing the concept of local and global minima and a discussion of the necessary and sufficient conditions for existence of a solution.

Local Minimum. A point $\boldsymbol{x}^{\star}$ is a local minimum of f if $f(\boldsymbol{x}^{*}) \leq f(\boldsymbol{x})$ in a neighborhood defined by $|\boldsymbol{x} - \boldsymbol{x}^{*}| < \delta$ for some $\delta > 0$.

Global Minimum. The point $\boldsymbol{x}^{\star}$ is a global minimum if $f(\boldsymbol{x}^{*}) \leq f(\boldsymbol{x}), \boldsymbol{x} \in \Omega,$ where Ω is the feasible region. Further, the point $\boldsymbol{x}^{\star}$ is a strong global minimum if: $f(\boldsymbol{x}^{*}) < f(\boldsymbol{x}), \boldsymbol{x} \in \Omega$.

The local and global minima are synonymous in the case of convex optimization problems. In the remaining cases, a distinction between the two needs to be made. Further, local or global minimum in the case of non-convex optimization problems is not necessarily unique.

Necessary and Sufficient Conditions. The conditions that *must* be satisfied at the optimum point are termed as necessary conditions. However, the set of points that satisfies the necessary conditions further includes maxima and points of inflection. The sufficient conditions are then used to qualify the solution point as an optimum point. If a candidate point satisfies the sufficient conditions, then it is indeed the optimum point.

We now proceed to derive the first and second order conditions of optimality in the case of unconstrained optimization problems.

Download free eBooks at bookboon.com

4.2.1 First Order Necessary Conditions (FONC)

We consider a multi-variable function $f(\boldsymbol{x}) = f(x_1, x_2, \dots, x_n)$ and wish to investigate the behavior of a candidate point $\boldsymbol{x}^*$. By definition, the point $\boldsymbol{x}^*$ is a local minimum of $f(\boldsymbol{x})$ only if $f(\boldsymbol{x}^*) \leq f(\boldsymbol{x})$ only if in the neighborhood of $\boldsymbol{x}^*$. To proceed, let $\delta\boldsymbol{x} = \boldsymbol{x} - \boldsymbol{x}^*$ define a small neighborhood around $\boldsymbol{x}^*$, then we may use first-order Taylor series expansion of f given as: $f(\boldsymbol{x}) = f(\boldsymbol{x}^*) + \nabla f(\boldsymbol{x}^*)^T \delta\boldsymbol{x}$ to express the condition for local minimum as:

$$\delta f = \nabla f(\boldsymbol{x}^*)^T \delta\boldsymbol{x} \geq 0 \tag{4.2}$$

We first note that the above condition is satisfied for $\nabla f(\boldsymbol{x}^*) = 0$. Further, since $\delta\boldsymbol{x}$ is arbitrary, $\nabla f(\boldsymbol{x}^*)$ must be zero to satisfy the above non-negativity condition on δf. Therefore, the first-order necessary condition (FONC) for optimality of $f(\boldsymbol{x})$ is stated as follows:

FONC: *If $f(\boldsymbol{x})$ has a local minimum at $\boldsymbol{x}^*$, then $\nabla f(\boldsymbol{x}^*) = 0$, or equivalently, $\frac{\partial f(x^*)}{\partial x_j} = 0,\ j = 1, \dots, n.$*

The points that satisfy FONC are called stationary points of $f(\boldsymbol{x})$. Besides minima, these points include maxima and the points of inflection.

Download free eBooks at bookboon.com

4.2.2 Second Order Conditions (SOC)

Assume now that FONC are satisfied, i.e., $\nabla f(\boldsymbol{x}^*) = 0$. Then, we may use second-order Taylor series expansion of $f(\boldsymbol{x})$ to write the optimality condition as:

$$\delta f = \delta \boldsymbol{x}^T \nabla^2 f(\boldsymbol{x}^*) \delta \boldsymbol{x} \geq 0 \tag{4.3}$$

As $\delta \boldsymbol{x}$ are arbitrary, the above quadratic form is positive (semi)definite if and only if the Hessian matrix, $\nabla^2 f(\boldsymbol{x}^*)$, is positive (semi)definite. Therefore, the second order necessary condition (SONC) is stated as:

SONC: *If* $\boldsymbol{x}^*$ *is a local minimizer of* $f(\boldsymbol{x})$ *, then* $\nabla^2 f(\boldsymbol{x}^*) \geq 0$.

Whereas, a second order sufficient condition (SOSC) is stated as:

SOSC: *If* $\boldsymbol{x}^*$ *satisfies* $\nabla^2 f(\boldsymbol{x}^*) > 0$, *, then* $\boldsymbol{x}^*$ *is a local minimizer of* $f(\boldsymbol{x})$.

Further, if $\nabla^2 f(\boldsymbol{x}^*)$ is indefinite, then $\boldsymbol{x}^*$ is an inflection point. In the event that $\nabla f(\boldsymbol{x}^*) = \nabla^2 f(\boldsymbol{x}^*) = 0$, , the lowest nonzero derivative must be even-ordered for stationary points (necessary condition), and it must be positive for local minimum (sufficient condition).

Two examples of unconstrained optimization problems are now considered:

Example 4.1: Polynomial data-fitting

As an example of unconstrained optimization, we consider the polynomial data-fitting problem defined in Sec. 2.11. The problem is to fit an th degree polynomial: $p(x) = \sum_{j=0}^{n} p_j x^j$ to a set of data points: $(x_i, y_i), i = 1, \dots, N > n$. The objective is to minimize the mean square error (MSE, also termed as the variance of data points). The resulting unconstrained minimization problem is formulated as:

$$\min_{p_j} f(p_j) = \frac{1}{2N} \sum_{i=1}^{N} \left(y_i - (p_0 + p_1 x_i + \cdots + p_n x_i^n) \right)^2$$

Then, the FONC for the problem are given as:

$$\frac{\partial f}{\partial p_j} = \frac{1}{N} \sum_{i=1}^{N} \left(y_i - (p_0 + p_1 x_i + \cdots + p_n x_i^n) \right) (-x_i^j) = 0$$

For $n = 1$, they result in the following equations:

$$\begin{pmatrix} 1 & \frac{1}{N}\sum_i x_i \\ \frac{1}{N}\sum_i x_i & \frac{1}{N}\sum_i x_i^2 \end{pmatrix} \begin{pmatrix} p_0 \\ p_1 \end{pmatrix} = \begin{pmatrix} \frac{1}{N}\sum_i y_i \\ \frac{1}{N}\sum_i x_i y_i \end{pmatrix}$$

Download free eBooks at bookboon.com

As $\frac{\partial^2 f}{\partial p_j p_k} = \frac{1}{N}\sum_i x_i^{j+k}$, the SONC for the problem are evaluated as:

$$\begin{pmatrix} 1 & \cdots & \frac{1}{N}\sum_i x_i^n \\ \vdots & \ddots & \vdots \\ \frac{1}{N}\sum_i x_i^n & \cdots & \frac{1}{N}\sum_i x_i^{2n} \end{pmatrix} \geq 0.$$

For $n = 1$, the determinant of the Hessian evaluates as: $\frac{1}{N}\sum_i x_i^2 - \left(\frac{1}{N}\sum_i x_i\right)^2$, which defines the variance in the case of independent and identically distributed random variables.

Finally, we note that since the data-fitting problem is convex, FONC are both necessary and sufficient for a minimum.

Example 4.2: Open box problem

We wish to determine the dimensions of an open box of maximum volume that can be constructed form a sheet of paper (8.5 ×11 in) by cutting squares from the corners and folding the sides upwards.

Let x denote the width of the paper that is folded up, then the problem is formulated as: $\max_x f(x) = (11 - 2x)(8.5 - 2x)x$

The FONC for the problem evaluate as: $f'(x) = 2x(19.5 - 4x) - (11 - 2x)(8.5 - 2x) = 0$.

Using Matlab Symbolic toolbox 'solve' command, we obtain two candidate solutions: $x^* = 1.585, 4.915$.

Application of SOC results in: $f''(x) = -39.95, 39.95$, respectively, indicating a maximum of $f(\boldsymbol{x})$ at $x^* = 1.585$ with $f(x^*) = 66.15$ cu.in.

4.3 Optimality Criteria for the Constrained Problems

The majority of engineering design problems involve constraints (LE, GE, EQ) that are modeled as functions of optimization variables. In this section, we explore how constraints affect the optimality criteria. An important consideration when applying the optimality criteria to problems involving constraints is whether $\boldsymbol{x}^*$ lies on a constraint boundary. This is implied in the case for problems involving only equality constraints, which are discussed first.

4.3.3 Equality Constrained Problems

The optimality criteria for equality constrained problems involve the use of Lagrange multipliers. To develop this concept, we consider a problem with a single equality constraint, stated as:

$$\begin{aligned} &\min_{\boldsymbol{x}} f(\boldsymbol{x}) \\ &\text{subject to } h(\boldsymbol{x}) = 0 \end{aligned} \tag{4.4}$$

Download free eBooks at bookboon.com

We first note that the constraint equation can be used to solve for and substitute one of the variables (say x_n) in the objective function, and hence develop an unconstrained optimization problem in variables. n-1 This, however, depends on the form of $h(\boldsymbol{x})$ and may not always be feasible. In order to develop more general optimality criteria, we follow Lagrange's approach to the problem and consider the variation in the objective and constraint functions at a stationary point, given as:

$$df = \frac{\partial f}{\partial x_1} dx_1 + \cdots + \frac{\partial f}{\partial x_n} dx_n = 0 \tag{4.5}$$

$$dh = \frac{\partial h}{\partial x_1} dx_1 + \cdots + \frac{\partial h}{\partial x_n} dx_n = 0$$

We now combine these two conditions via a scalar weight (Lagrange multiplier, λ) to write:

$$\sum_{j=1}^{n} \left(\frac{\partial f}{\partial x_j} + \lambda \frac{\partial h}{\partial x_j} \right) dx_j = 0 \tag{4.6}$$

Since variations dx_j are independent, the above condition implies that: $\frac{\partial f}{\partial x_j} + \lambda \frac{\partial h}{\partial x_j} = 0,\ j = 1, \ldots, n.$

Download free eBooks at bookboon.com

We further note that application of FONC to a Lagrangian function defined as: $\mathcal{L}(\boldsymbol{x}, \lambda) = f(\boldsymbol{x}) + \lambda h(\boldsymbol{x})$ also gives rise to the above condition. For multiple equality constraints, the Lagrangian function is similarly formulated as:

$$\mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}) = f(\boldsymbol{x}) + \sum_{i=1}^{l} \lambda_i h_i(\boldsymbol{x}) \tag{4.7}$$

Then, in order for $f(\boldsymbol{x})$ to have a local minimum at $\boldsymbol{x}^\star$, the following FONC must be satisfied:

$$\frac{\partial \mathcal{L}}{\partial x_j} = \frac{\partial f}{\partial x_j} + \sum_{i=1}^{l} \lambda_i \frac{\partial h_i}{\partial x_j} x_j = 0; \;\; j = 1, \dots, n \tag{4.8}$$

$$h_i(\boldsymbol{x}) = 0, \;\; i = 1, \dots, l$$

The above FONC can be equivalently stated as:

$$\nabla_{\boldsymbol{x}} \mathcal{L}(\boldsymbol{x}^*, \boldsymbol{\lambda}^*) = 0, \qquad \nabla_{\boldsymbol{\lambda}} \mathcal{L}(\boldsymbol{x}^*, \boldsymbol{\lambda}^*) = 0$$

These conditions suggest that $\mathcal{L}(\boldsymbol{x}^*, \boldsymbol{\lambda}^*)$ is stationary with respect to both $\boldsymbol{x}$ and $\boldsymbol{\lambda}$; therefore, minimization of $\mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda})$ amounts to an unconstrained optimization problem. Further, the Lagrange Multiplier Theorem (Arora, p.135) states that if $\boldsymbol{x}^\star$ is a regular point (defined below) then the FONC result in a unique solution to λ_i^*.

We note that the above FONC further imply: $\nabla f(\boldsymbol{x}^*) = -\sum_{i=1}^{p} \lambda_i \nabla h_i(\boldsymbol{x}^*)$. Algebraically, it means that the cost function gradient is a linear combination of the constraint gradients. Geometrically, it means that the negative of the cost function gradient lies in the convex cone spanned by the constraint normals (Sec. 4.3.5).

SOSC for equality constrained problems are given as: $\nabla^2 \mathcal{L}(\boldsymbol{x}^*, \boldsymbol{\lambda}^*) = \nabla^2 f(\boldsymbol{x}^*) > 0$. Further discussion on SOC for constrained optimization problems is delayed till Sec. 4.4.3.

An example is now presented to explain the optimization process for equality constrained problems.

Example 4.3: We consider the following optimization problem:

$$\min_{x_1, x_2} \; f(x_1, x_2) = -x_1 x_2$$

Subject to: $h(x_1, x_2) = x_1^2 + x_2^2 - 1 = 0$

Download free eBooks at bookboon.com

We first note that the equality constraint can be used to develop an unconstrained problem in one variable, given as: $\min_{x_1} f(x_1) = -x_1\sqrt{1-x_1^2}$ or $\min_{x_2} f(x_2) = -x_2\sqrt{1-x_2^2}$. or . Instead, we follow the Lagrangian approach to solve the original problem below.

Next, we assess the objective function and observe that the origin is saddle point of the function. It is also instructive to review the problem from a graphical perspective (Figure 4.1). The figure shows the feasible region, i.e., the perimeter of a unit circle superimposed on the level sets of the objective function. Then, by inspection, the optimum can be located in the first and the third quadrant where the level curves are tangent to the circle.

The Lagrangian function for the problem is formulated as: $\mathcal{L}(x_1, x_2, \lambda) = -x_1x_2 + \lambda(x_1^2 + x_2^2 - 1)$.
The FONC evaluate as: $2\lambda x_1 - x_2 = 0,\ 2\lambda x_2 - x_1 = 0,\ x_1^2 + x_2^2 - 1 = 0$.
Thus, there are four candidate solutions at: $(x_1^*, x_2^*) = \left(\pm\frac{1}{\sqrt{2}}, \pm\frac{1}{\sqrt{2}}\right), \lambda^* = \pm\frac{1}{2}$.

The SONC for the problem evaluate as: $\begin{bmatrix} 2\lambda & -1 \\ -1 & 2\lambda \end{bmatrix} \geq 0$. Application of SONC reveals multiple minima at $(x_1^*, x_2^*) = \left(\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}\right) \cup \left(-\frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}}\right)$ with $f(x_1^*, x_2^*) = -\frac{1}{2}$.

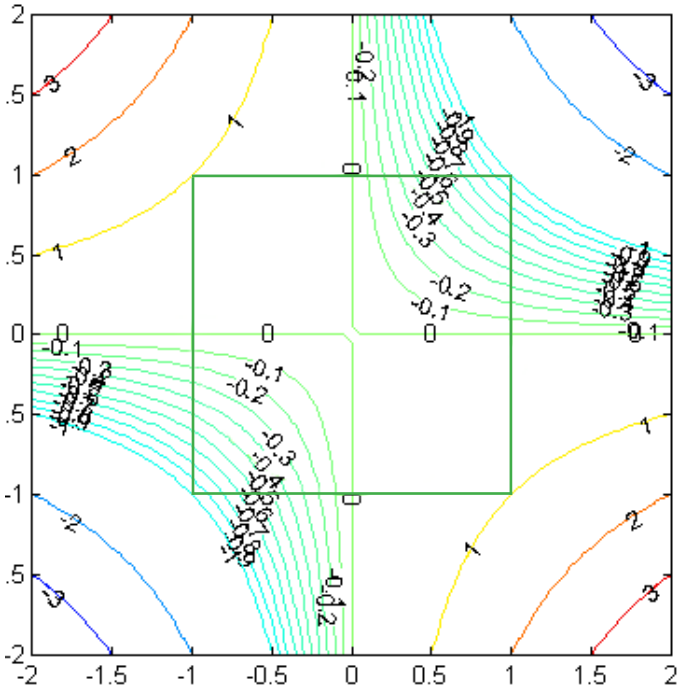


Figure 4.1: Level sets of the objective function superimposed on the equality constraint.

Download free eBooks at bookboon.com

The above example underscores some of the pitfalls in the case of nonlinear optimization problems: Application of FONC results in multiple nonlinear equations whose simultaneous solution reveals several candidate points, which pertain to maxima, minima and points of inflection. The minima may then be obtained via application of SOSC or via a comparison of function values at the individual points.

4.3.4 Inequality Constrained Problems

We next consider an optimization problem involving a single inequality constraint. The problem is stated as:

$$\begin{aligned} &\min_{\boldsymbol{x}} \ f(\boldsymbol{x}) \\ &\text{subject to } g(\boldsymbol{x}) \leq 0 \end{aligned} \tag{4.10}$$

We note that we can add a slack variable to the inequality constraint to turn it into equality. Further, to ensure constraint compliance, the slack variable is restricted to be non-negative. We therefore replace the inequality constraint with equality: $g(\boldsymbol{x}) + \mathrm{s}^2 = 0$. A Lagrangian function for the problem is now developed as:

$$\mathcal{L}(\boldsymbol{x}, \lambda, s) = f(\boldsymbol{x}) + \lambda(g(\boldsymbol{x}) + \mathrm{s}^2) \tag{4.11}$$

Download free eBooks at bookboon.com

The resulting FONC evaluate as:

$$\begin{aligned}&\nabla\mathcal{L}(\boldsymbol{x}^*,\lambda^*,s^*)=\nabla f(\boldsymbol{x}^*)+\lambda^*\nabla g(\boldsymbol{x}^*)=0\\&g(\boldsymbol{x}^*)+\mathrm{s}^{*2}=0\\&\frac{\partial\mathcal{L}}{\partial s}=2\lambda^*s^*=0\end{aligned}\tag{4.12}$$

The latter condition, known as the switching or complementarity condition, further evaluates as: $\lambda = 0$ (thus implying an inactive constraint) or $s = 0$ (implying an active/binding constraint). Each of these cases is to be explored for feasible solutions, which can be checked for optimality via application of SOC.

We note that by substituting: $\mathrm{s}^2 = -g(\boldsymbol{x}^*)$, the FONC can be equivalently expressed as: $\nabla f(\boldsymbol{x}^*) = 0$, $g(\boldsymbol{x}^*) \leq 0$, $\lambda^* g(\boldsymbol{x}^*) = 0$, which provides an equivalent characterization of FONC in the case of inequality constrained problems.

Finally, the above results can be extended to multiple inequality constraints as:

$$\mathcal{L}(\boldsymbol{x},\lambda,s)=f(\boldsymbol{x})+\sum_i\lambda_i\left(g_i(\boldsymbol{x})+s_i^2\right)\tag{4.13}$$

Then, in order for $f(\boldsymbol{x})$ to have a local minimum at $\boldsymbol{x}^*$, the following FONC must be satisfied:

$$\begin{aligned}&\frac{\partial\mathcal{L}}{\partial x_j}=\frac{\partial f}{\partial x_j}+\sum\nolimits_{i=1}^{m}\lambda_i\frac{\partial g_i}{\partial x_j}x_j=0;\ \ j=1,\dots,n\\&g_i(\boldsymbol{x})+s_i^2=0,\ \ i=1,\dots,m\\&\lambda_i s_i=0,i=1,\dots,m\end{aligned}\tag{4.14}$$

where we note that for m inequality constraints, application of the switching conditions results in cases, each of which needs to be explored for feasibility and optimality.

Next, we present an example of the inequality constrained problem.

Example 4.4: We consider the following optimization problem:

$$\begin{aligned}&\min_{x_1,x_2}\ f(x_1,x_2)=-x_1x_2\\&\text{Subject to: } g(x_1,x_2)=x_1^2+x_2^2-1\leq 0\end{aligned}$$

The graphical consideration of the equality constrained problem was earlier presented in Fig. 4.1. From that figure, it is obvious that the inequality constrained problem will have a solution at the boundary of the constraint set, i.e., at the perimeter of the circle. This view is supported by the analysis presented below.

We first convert the inequality to equality constraint via: $g(x_1,x_2)+s^2=x_1^2+x_2^2-1+s^2=0$.

Download free eBooks at bookboon.com

Then, the Lagrangian function is formulated as: $\mathcal{L}(x_1, x_2, \lambda, s) = -x_1 x_2 + \lambda(x_1^2 + x_2^2 + s^2 - 1)$.
The resulting FONC evaluate as: $2\lambda x_1 - x_2 = 0,\ 2\lambda x_2 - x_1 = 0,\ x_1^2 + x_2^2 + s^2 - 1 = 0,\ \lambda s = 0$.

The switching condition further evaluates as: $\lambda^* = 0$ or $s^* = 0$. The former condition evaluates as: $(x_1^*, x_2^*) = (0,0),\ s^* = \pm 1$, while latter condition evaluates as: $(x_1^*, x_2^*) = \left(\pm\frac{1}{\sqrt{2}}, \pm\frac{1}{\sqrt{2}}\right), \lambda^* = \pm\frac{1}{2}$.

Function evaluation at the candidate points reveals multiple minima at $(x_1^*, x_2^*) = \left(\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}\right) \cup \left(-\frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}}\right)$ with $f(x_1^*, x_2^*) = -\frac{1}{2}$.

4.4 Optimality Criteria for General Optimization Problems

The general nonlinear optimization problem was defined in (4.1) above, where we can group the variable limits with the inequality constraints to state the problem as:

$$\min_{\boldsymbol{x}} f(\boldsymbol{x}) \tag{4.15}$$

Subject to: $h_i(\boldsymbol{x}) = 0,\ i = 1, \dots, l;\ g_j(\boldsymbol{x}) \leq 0, j = i, \dots, m$

The feasible region for the problem is given as:

$$\Omega = \left\{\boldsymbol{x}: h_i(\boldsymbol{x}) = 0, i = 1, \dots, l;\ g_j(\boldsymbol{x}) \leq 0, j = 1, \dots, m\right\} \tag{4.16}$$

To solve the problem via the Lagrangian function approach, we first add slack variables to the inequality constraints; we then associate Lagrange multiplier vectors $\boldsymbol{u}$ and $\boldsymbol{v}$ with the inequality and equality constraints, respectively, and develop a Lagrangian function, which is given as:

$$\mathcal{L}(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{v}, \boldsymbol{s}) = f(\boldsymbol{x}) + \sum_{i=1}^{l} v_i h_i(\boldsymbol{x}) + \sum_{j=1}^{m} u_j (g_j(\boldsymbol{x}) + s_j^2) \tag{4.17}$$

The resulting FONC evaluate as:

1. Gradient conditions: $\frac{\partial \mathcal{L}}{\partial x_k} = \frac{\partial f}{\partial x_k} + \sum_{i=1}^{l} v_i^* \frac{\partial h_i}{\partial x_k} + \sum_{j=1}^{m} u_j^* \frac{\partial g_j}{\partial x_k} = 0;\ k = 1, \dots, n$
2. Switching conditions: $u_j^* s_j = 0,\ j = 1, \dots, m$
3. Feasibility conditions: $g_j(\boldsymbol{x}^*) \leq 0,\ j = 1, \dots, m;\ h_i(\boldsymbol{x}) = 0,\ i = 1, \dots, p$
4. Non-negativity condition: $u_j^* \geq 0,\ j = 1, \dots, m$
5. Regularity condition: for those u_j^*, that satisfy $u_j^* > 0,\ \nabla g_j(\boldsymbol{x}^*)$ are linearly independent

The above FONC are better known as the KKT (Krush-Kuhn-Tucker) conditions.

Download free eBooks at bookboon.com

We note that $\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{v}, \boldsymbol{s}$ are, respectively, n-, m-, m-, and l-dimensional vectors. Thus, the total number of variables in the problem is: $n + 2m + l$, meaning simultaneous nonlinear equations must be solved to obtain a candidate solution. Further, in accordance with the switching conditions, a total of 2^m such solutions must be explored.

Further, since $s_j^2 = -g_j(\boldsymbol{x})$, non-negativity of s_j^2 ensures feasibility of the inequality constraint. Therefore $s_j^2 = 0$, implies an active constraint, whereby $u_j^* > 0$; and an inactive constraint is implied by: $u_j^* = 0, s_j^2 > 0$. We also note that for regular points the Lagrange Multiplier Theorem (Arora, p.135) ensures a unique solution to the Lagrange multipliers v_i^* and u_j^*.

An example of general optimization problem is presented below.

Example 4.5: We consider adding a linear equality constraint to Example 4.4 above:

$$\min_{x_1,x_2} f(x_1, x_2) = -x_1 x_2$$
Subject to: $g(x_1, x_2): x_1^2 + x_2^2 - 1 \leq 0; \;\; h(x_1, x_2): x_1 + x_2 - c = 0$

We first convert the inequality to equality constraint via: $g(x_1, x_2) + s^2 = x_1^2 + x_2^2 - 1 + s^2 = 0$.

Download free eBooks at bookboon.com

We then use Lagrange multipliers to formulate a Lagrangian function, which is given as: $\mathcal{L}(x_1, x_2, u, v, s) = -x_1x_2 + u(x_1^2 + x_2^2 + s^2 - 1) + v(x_1 + x_2 - c)$.

The resulting KKT conditions evaluate as: $2ux_1 + v - x_2 = 0,\ 2ux_2 + v - x_1 = 0,\ x_1 + x_2 - c = 0,$ $x_1^2 + x_2^2 + s^2 - 1 = 0,\ us = 0$. From the switching condition: or $u^* = 0$ or $s^* = 0$.

The former condition evaluates as: $(x_1^*, x_2^*) = \left(\frac{c}{2}, \frac{c}{2}\right),\ s^* = \pm\sqrt{1 - \frac{c^2}{2}}, v^* = \frac{c}{2}$.

The latter condition has no feasible solution.

Function evaluation at the sole candidate points results in: $f(x_1^*, x_2^*) = -\frac{c^2}{4}$.

4.4.1 Optimality Criteria for Convex Optimization Problems

In this section we consider the class of optimization problems where the feasible region is a convex set and the objective and constraint functions are convex. The convexity property is desirable since it implies the existence of a global minimum to the optimization problem.

We consider the general optimization problem defined in (4.15) with the feasible region given by (4.16). Then Ω, is a convex set if functions h_i are linear and g_j are convex. If additionally $f(\boldsymbol{x})$ is a convex function, then the optimization problem is convex.

We now assume that $f(\boldsymbol{x})$ is a convex function defined over a convex set Ω. Then, if $f(\boldsymbol{x})$ attains a local minimum at $\boldsymbol{x}^* \in \Omega$, then $\boldsymbol{x}^\star$ is also a global minimum over Ω. Furthermore, $f(\boldsymbol{x}^*)$ is a local/global minimum if and only if it satisfies the KKT conditions, i.e., the KKT conditions are both necessary and sufficient for a global minimum in the case of convex optimization problems.

We, however, note that convexity is a sufficient but not necessary condition for a global minimum, i.e., nonexistence of convexity does not preclude the existence of a global minimum. An example of a convex optimization problem is presented below.

Example 4.6: We consider the following optimization problem:

$\min_{x_1,x_2}\ f(x_1, x_2) = x_1^2 + x_2^2 - x_1x_2,$
subject to $g(x_1, x_2)$: $x_1^2 + x_2^2 - 1 \leq 0;\ h(x_1, x_2)$: $x_1 + x_2 - c = 0$

As was done in Example 4.5, we convert the inequality constraint to equality via: $x_1^2 + x_2^2 - 1 + s^2 = 0$. We then use Lagrange multipliers to formulate a Lagrangian function given as:

$\mathcal{L}(x_1, x_2, u, v, s) = x_1^2 + x_2^2 - x_1x_2 + u(x_1^2 + x_2^2 + s^2 - 1) + v(x_1 + x_2 - c)$.

Download free eBooks at bookboon.com

The resulting KKT conditions evaluate as: $(2u+1)x_1 + v - x_2 = 0,\ (2u+1)x_2 + v - x_1 = 0,\ x_1 + x_2 - c = 0,\ x_1^2 + x_2^2 + s^2 - 1 = 0,\ us = 0$. From the switching condition: $u^* = 0$ or $s^* = 0$.

Similar to Example 4.5, the former condition evaluates as: $(x_1^*, x_2^*) = \left(\frac{c}{2}, \frac{c}{2}\right),\ s^* = \pm\sqrt{1 - \frac{c^2}{2}}, v^* = \frac{c}{2}$; while the latter condition has no feasible solution. Function evaluation at the sole candidate points results in: $f(x_1^*, x_2^*) = -\frac{c^2}{4}$.

4.4.2 A Geometric Viewpoint

The optimality criteria for constrained optimization problems have geometrical connotations. The following definitions help understand the geometrical viewpoint associated with the KKT conditions.

Active constraint set. The set of active constraints at $\boldsymbol{x}$ is defined as: $\mathcal{I} = \{i \cup j : h_i(\boldsymbol{x}) = 0, g_j(\boldsymbol{x}) = 0\}$. The set of active constraint normals is given as: $\mathcal{S} = \{\nabla h_i(\boldsymbol{x}), \nabla g_j(\boldsymbol{x}), j \in \mathcal{I}\}$.

Constraint tangent hyperplane. The constraint tangent hyperplane is defined by the set of vectors $\mathcal{S}^{\perp} = \{\boldsymbol{d} : \nabla h_i(\boldsymbol{x})^T \boldsymbol{d} = 0,\ \nabla g_j(\boldsymbol{x})^T \boldsymbol{d} = 0, j \in \mathcal{I}\}$.

Regular point. Assume $\boldsymbol{x}$ is a feasible point. Then, $\boldsymbol{x}$ is a regular point if the vectors in the active constraint set $\mathcal{S}$ are linearly independent.

Feasible direction. Assume that $\boldsymbol{x}$ is a regular point. A vector $\boldsymbol{d}$ is a feasible direction if $h_i(\boldsymbol{x})^T \boldsymbol{d} = 0$, $\nabla g_j(\boldsymbol{x})^T \boldsymbol{d} < 0, j \in \mathcal{I}$; where the feasibility condition for each active inequality constraint defines a half space. The intersection of those half spaces is a feasible cone within which a feasible vector $\boldsymbol{d}$ should lie.

Descent direction. A direction $\boldsymbol{d}$ is a descent direction if the directional derivative of f along $\boldsymbol{d}$ is negative, i.e., $\nabla f(\boldsymbol{x})^T \boldsymbol{d} < 0$.

Extreme point. Assume $\boldsymbol{x}$ is a feasible point. Then, $\boldsymbol{x}$ is an extreme point if the active constraint set $\mathcal{I}$ at $\boldsymbol{x}$ is non-empty; otherwise it is an interior point.

Assume now that we are at an extreme point $\boldsymbol{x}$ of the feasible region. We seek a search direction which is both descent and feasible. If no such direction can be found then we have already reached the optimum. Geometrical categorization of the optimal point rests on the following lemma.

Farka's Lemma. For $\boldsymbol{A} \in \mathbb{R}^{n \times m},\ \boldsymbol{c} \in \mathbb{R}^n$ only one of the two problems has a solution:

1. $\boldsymbol{A}^T \boldsymbol{x} \geq \boldsymbol{0}, \boldsymbol{c}^T \boldsymbol{x} < 0$
2. $\boldsymbol{c} = \boldsymbol{A}\boldsymbol{y}, \boldsymbol{y} \geq \boldsymbol{0}$

Download free eBooks at bookboon.com

Corollary. For any $\boldsymbol{A} \in \mathbb{R}^{n\times m}$, $\boldsymbol{c} \in \mathbb{R}^{n}$, we have $\boldsymbol{A}^T\boldsymbol{x} \geq \boldsymbol{0}, \boldsymbol{c}^T\boldsymbol{x} \geq 0$ if and only if $\boldsymbol{c} = \boldsymbol{A}\boldsymbol{y}, \boldsymbol{y} \geq \boldsymbol{0}$.

Farka's lemma was used in the proof of Karush-Kuhn-Tucker (KKT) Theorem on NLP by Tucker. The lemma states that if a vector $\boldsymbol{c}$ does not lie in the convex cone: $\boldsymbol{C} = \{\boldsymbol{A}\boldsymbol{y}, \boldsymbol{y} \geq \boldsymbol{0}\}$, then there is a vector $\boldsymbol{x}, \boldsymbol{A}^T\boldsymbol{x} \geq \boldsymbol{0}$, that is normal to a hyperplane separating $\boldsymbol{c}$ from $\boldsymbol{C}$.

To apply this lemma we consider a matrix $\boldsymbol{A}$ whose columns represent the active constraint gradients at the optimum point x^*, a vector that represents the objective function gradient $\nabla f(\boldsymbol{x}^*)$, and $\boldsymbol{d}$ represents a search direction. Then, there is no $\boldsymbol{d}$ satisfying the descent and feasibility conditions: $\nabla f(\boldsymbol{x}^*)^T\boldsymbol{d} < 0$ and $\nabla g_j(\boldsymbol{x})^T\boldsymbol{d} > 0, j \in \mathcal{J}$, if and only if the objective function gradient can be expressed as: $-\nabla f(\boldsymbol{x}^*) = \sum_{j\in\mathcal{J}} \lambda_j \nabla g_j(\boldsymbol{x}^*), \lambda_j \geq 0$.

The above lemma also explains the non-negativity condition on Lagrange multipliers for inequality constraints in the KKT conditions.

Download free eBooks at bookboon.com

4.4.3 Second Order Conditions

The second order necessary and sufficient conditions assume that $\boldsymbol{x}^{\star}$ satisfies the FONC (the KKT conditions) and use the Hessian of the Lagrangian function to investigate the behavior of the candidate point $\boldsymbol{x}^{\star}$ where we. The Hessian of the Lagrangian is defined as:

$$\nabla^2 \mathcal{L}(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{v}, \boldsymbol{s}) = \nabla^2 f(\boldsymbol{x}) + \sum_{i=1}^{l} v_i \nabla^2 h_i(\boldsymbol{x}) + \sum_{j=1}^{m} u_j \nabla^2 g_j(\boldsymbol{x}) \tag{4.18}$$

Second Order Necessary Condition (SONC). Assume $\boldsymbol{d}$ is a feasible vector that lies in the constraint tangent hyperplane, i.e., let $\{\boldsymbol{d}: \nabla h_i(\boldsymbol{x})^T \boldsymbol{d} = 0,\ \nabla g_j(\boldsymbol{x})^T \boldsymbol{d} = 0, j \in \mathcal{I}\}$. If $\boldsymbol{x}^*$ is a local minimizer of *f,* then it satisfies the following SONC:

$$\delta f = \boldsymbol{d}^T \nabla^2 \mathcal{L}(\boldsymbol{x}^*, \boldsymbol{u}^*, \boldsymbol{v}^*, \boldsymbol{s}^*) \boldsymbol{d} \geq 0 \tag{4.19}$$

Second Order Sufficient Condition (SOSC). If for some $\boldsymbol{x}^*$, $\boldsymbol{d}^T \nabla^2 \mathcal{L}(\boldsymbol{x}^*, \boldsymbol{u}^*, \boldsymbol{v}^*, \boldsymbol{s}^*) \boldsymbol{d} > 0$ for all $\{\boldsymbol{d}: \nabla h_i(\boldsymbol{x}^*)^T \boldsymbol{d} = 0,\ \nabla g_j(\boldsymbol{x}^*)^T \boldsymbol{d} = 0, u_j^* > 0\}$, then $\boldsymbol{x}^{\star}$ is a local minimizer of $f(\boldsymbol{x})$.

Further, a stronger SOSC is given as: $\nabla^2 \mathcal{L}(\boldsymbol{x}^*, \boldsymbol{u}^*, \boldsymbol{v}^*, \boldsymbol{s}^*) > 0$, then $\boldsymbol{x}^*$ is a local minimizer of $f(\boldsymbol{x})$.

An example of SOC is now presented.

Example: Second order conditions

We consider the optimization problem in Example 4.5. The constraint tangent hyperplane for active constraints in Example 4.5 is computed as: $d_1 - d_2 = 0$. The Hessian of the Lagrangian at the optimum point $(x_1^*, x_2^*) = \left(\frac{c}{2}, \frac{c}{2}\right)$ is given as: $\begin{pmatrix} 0 & -1 \\ -1 & 0 \end{pmatrix}$. Therefore, the SONC evaluate as: $\boldsymbol{d}^T \nabla^2 \mathcal{L}\, \boldsymbol{d} = 2$, indicating that the candidate point is indeed an optimum point. Similar analysis applies to Example 4.6.

4.5 Postoptimality Analysis

Postoptimality analysis refers to the study of the effects of parametric changes on the optimal solution. In particular, we are interested in the objective function variation resulting from relaxing the constraint limits. To study these changes, we consider the following perturbed optimization problem (Arora, p.153):

$$\begin{aligned} &\min_{\boldsymbol{x}} \ f(\boldsymbol{x}) \\ &\text{Subject to } \ h_i(\boldsymbol{x}) = b_i,\ i = 1, \ldots, l;\ \ g_j(\boldsymbol{x}) \leq e_j,\ j = i, \ldots, m \end{aligned} \tag{4.20}$$

Download free eBooks at bookboon.com

where b_i and e_j are small variations in the neighborhood of zero. Let the optimum point for the perturbed problem be expressed as: $\boldsymbol{x}^*(\boldsymbol{b},\boldsymbol{e})$, with the optimal cost given as: $f^*(\boldsymbol{b},\boldsymbol{e})$. Then, the implicit first order derivatives of the cost function are computed as: $\frac{\partial f(\boldsymbol{x}^*)}{\partial b_i} = -v_i^*$, $\frac{\partial f(\boldsymbol{x}^*)}{\partial e_j} = -u_j^*$; and, the resulting cost function variation due to constraint relaxation is given as:

$$\delta f(\boldsymbol{x}^*) = -\sum_i v_i^* b_i - \sum_j u_j^* e_j \tag{4.21}$$

The above result implies that the non-zero Lagrange multipliers accompanying the active constraints determine the cost function sensitivity to constraint relaxation. Non-active constraints have zero Lagrange multipliers, and hence do not affect the solution. Further, if the Lagrange multipliers for active constraints were to take on negative values, then constraint relaxation would result in a reduction in the optimum cost function value, which is counter-intuitive.

The cost function variation resulting from changes to parameters embedded in the constraints, $h_i(s)$ and $g_j(s)$, can be similarly examined by considering how individual constraint variations affect the cost function, i.e.,

$$\delta f(\boldsymbol{x}^*) = \sum_i v_i^* \delta h_i + \sum_j u_j^* \delta v_j \tag{4.22}$$

where, once again, we observe that Lagrange multipliers for the individual constraints determine the sensitivity of $\delta f(\boldsymbol{x}^*)$ to the parameter variations related to those constraints.

4.6 Lagrangian Duality

Lagrangian duality in optimization problems stems from the fact that the Lagrangian function is stationary at the optimal point. The duality theory associates with every optimization problem (termed as primal) a dual problem that uses Lagrange multipliers as the optimization variables.

To develop the duality concepts, we consider a general optimization problem (4.15), where a Lagrangian function for the problem was defined in (417). Equivalently, the Lagrangian function can be written without the slack variables, and in vector format the function and its derivatives are given as:

$$\begin{aligned} &\mathcal{L}(\boldsymbol{x},\boldsymbol{u},\boldsymbol{v}) = f(\boldsymbol{x}) + \boldsymbol{u}^T\boldsymbol{g} + \boldsymbol{v}^T\boldsymbol{h} \\ &\nabla\mathcal{L}(\boldsymbol{x},\boldsymbol{u},\boldsymbol{v}) = \nabla f(\boldsymbol{x}) + [\boldsymbol{\nabla g}]\boldsymbol{u} + [\boldsymbol{\nabla h}]\boldsymbol{v} \end{aligned} \tag{4.23}$$

where $[\boldsymbol{\nabla g}], [\boldsymbol{\nabla h}]$ are Jacobian matrices containing individual constraint derivatives as column vectors.

Download free eBooks at bookboon.com

Next, let $\boldsymbol{x}^{\star}$ represent an optimal solution to the problem and let $(\boldsymbol{u}^*, \boldsymbol{v}^*)$ be the associated Lagrange multipliers, then the Lagrangian function is stationary at the optimum point, i.e. $\nabla \mathcal{L}(\boldsymbol{x}^*, \boldsymbol{u}^*, \boldsymbol{v}^*) = 0$. To proceed further, we assume that the Hessian of the Lagrangian is positive definite in the feasible region, i.e., $\nabla^2 \mathcal{L}(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{v}) \geq \boldsymbol{0}, \ \boldsymbol{x} \in \Omega$. We can now state the following duality theorem (Belegundu and Chandrupatla, p. 269):

Duality theorem: The following are equivalent:

1. $\boldsymbol{x}^{\star}$ together with $(\boldsymbol{u}^*, \boldsymbol{v}^*)$ solves the primal problem (4.15).

2. $(\boldsymbol{x}^*, \boldsymbol{u}^*, \boldsymbol{v}^*)$ is a saddle point of the Lagrangian function $\mathcal{L}(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{v})$, i.e.,

$$\mathcal{L}(\boldsymbol{x}^*, \boldsymbol{u}, \boldsymbol{v}) \leq \mathcal{L}(\boldsymbol{x}^*, \boldsymbol{u}^*, \boldsymbol{v}^*) \leq \mathcal{L}(\boldsymbol{x}, \boldsymbol{u}^*, \boldsymbol{v}^*) \tag{4.24}$$

3. $(\boldsymbol{x}^*, \boldsymbol{u}^*, \boldsymbol{v}^*)$ solves the following dual problem:

$$\max_{\boldsymbol{x}, \boldsymbol{u} \geq \boldsymbol{0}, \boldsymbol{v}} \mathcal{L}(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{v}), \text{ Subject to } \nabla \mathcal{L}(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{v}) = 0 \tag{4.25}$$

Further, the two extrema are equal, i.e., $\mathcal{L}(\boldsymbol{x}^*, \boldsymbol{u}^*, \boldsymbol{v}^*) = f(\boldsymbol{x}^*)$

Download free eBooks at bookboon.com

In (4.24) above, $\mathcal{L}(\boldsymbol{x}^*, \boldsymbol{u}, \boldsymbol{v}) = \min_{\boldsymbol{x}\in\Omega} \mathcal{L}(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{v})$ represents a minimizer of $\mathcal{L}$ when $\boldsymbol{u} \geq \boldsymbol{0}, \boldsymbol{v}$ are fixed; similarly, $\mathcal{L}(\boldsymbol{x}, \boldsymbol{u}^*, \boldsymbol{v}^*) = \max_{\boldsymbol{u}\geq\boldsymbol{0},\boldsymbol{v}} \mathcal{L}(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{v})$ is a maximizer of $\mathcal{L}$ when $\boldsymbol{x} \in \Omega$ is fixed. These two functions, respectively, lower and upper bound the Lagrangian at the optimum point. Hence, $f(\boldsymbol{x}^*) \leq f(\boldsymbol{x})$ for any $\boldsymbol{x}$ that is primal-feasible, and $\mathcal{L}(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{v}) \leq f(\boldsymbol{x}^*)$ for any $\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{v}$ that are dual feasible $(\nabla\mathcal{L} = 0, \boldsymbol{u} \geq \boldsymbol{0})$.

Further, $\max_{\boldsymbol{u}\geq\boldsymbol{0},\boldsymbol{v}} \mathcal{L}(\boldsymbol{x}^*, \boldsymbol{u}, \boldsymbol{v}) \leq \min_{\boldsymbol{x}\in\Omega} \mathcal{L}(\boldsymbol{x}, \boldsymbol{u}^*, \boldsymbol{v}^*)$, which signifies weak duality. As a consequence of duality, if the dual objective is unbounded then primal problem has no feasible solution and vice versa.

We note that in nonlinear problems achieving strong duality (equality in 4.24) is not always possible, and, in general, a duality gap exists between the primal and dual solutions. Nevertheless, existence of strong duality is ensured in the case of convex optimization problems that satisfy the positive definite assumption on the Hessian matrix. Those problems are discussed following the dual function formulation.

4.6.1 Formulation of the Dual Function

The definition of the dual problem in (4.24) assumes that $\nabla^2\mathcal{L}(\boldsymbol{x}^*, \boldsymbol{u}^*, \boldsymbol{v}^*)$ is positive definite. If this assumption is valid, then by Implicit Function Theorem there exists a neighborhood $\mathcal{X}$ around $(\boldsymbol{x}^*, \boldsymbol{u}^*, \boldsymbol{v}^*)$ in which $\boldsymbol{x} = \boldsymbol{x}(\boldsymbol{u}, \boldsymbol{v})$ is a differentiable function, $\nabla\mathcal{L}(\boldsymbol{x}(\boldsymbol{u}, \boldsymbol{v}), \boldsymbol{u}, \boldsymbol{v}) = 0$, and $\nabla^2\mathcal{L}(\boldsymbol{x}(\boldsymbol{u}, \boldsymbol{v}), \boldsymbol{u}, \boldsymbol{v})$, and is positive definite. Moreover, every $(\boldsymbol{u} \geq \boldsymbol{0}, \boldsymbol{v})$ close to $(\boldsymbol{u}^*, \boldsymbol{v}^*)$ close to has a unique corresponding $\boldsymbol{x}$ that is a global minimizer of $\mathcal{L}(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{v})$ in $\mathcal{X}$. This allows us to define a dual function $\varphi(\boldsymbol{u}, \boldsymbol{v})$ as:

$$\varphi(\boldsymbol{u}, \boldsymbol{v}) = \min_{\boldsymbol{x}\in\mathcal{X}} \mathcal{L}(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{v}) \tag{4.26}$$

where the minimum can be found from the application of FONC. In terms of the dual function the dual optimization problem is defined as:

$$\text{(D)} \max_{\boldsymbol{u}\geq\boldsymbol{0},\boldsymbol{v}} \varphi(\boldsymbol{u}, \boldsymbol{v}) \tag{4.27}$$

Let $(\boldsymbol{u}^*, \boldsymbol{v}^*)$ be the optimal solution to the dual problem, then $\varphi(\boldsymbol{u}^*, \boldsymbol{v}^*) = f(\boldsymbol{x}^*)$.

The dual problem may be similarly solved via application of FONC. Towards this end, we use the chain rule to compute the derivative of the dual function as (Griva, Nash & Sofer, p. 537):

$$\nabla\varphi(\boldsymbol{u}, \boldsymbol{v}) = \nabla_{\boldsymbol{u},\boldsymbol{v}}\boldsymbol{x}(\boldsymbol{u}, \boldsymbol{v})\, \nabla_x\varphi(\boldsymbol{u}, \boldsymbol{v}) + \nabla_{\boldsymbol{u},\boldsymbol{v}}\varphi(\boldsymbol{u}, \boldsymbol{v}) \tag{4.28}$$

Download free eBooks at bookboon.com

where, by definition, $\nabla_x \varphi(\boldsymbol{u}, \boldsymbol{v}) = 0$. Further, $\nabla_{\boldsymbol{u}} \varphi(\boldsymbol{u}, \boldsymbol{v}) = \boldsymbol{g}\big(\boldsymbol{x}(\boldsymbol{u}, \boldsymbol{v})\big)$, $\nabla_{\boldsymbol{v}} \varphi(\boldsymbol{u}, \boldsymbol{v}) = \boldsymbol{h}\big(\boldsymbol{x}(\boldsymbol{u}, \boldsymbol{v})\big)$, and $\nabla_{\boldsymbol{u},\boldsymbol{v}} \varphi(\boldsymbol{u}, \boldsymbol{v}) = \begin{bmatrix} \boldsymbol{g}(\boldsymbol{x}(\boldsymbol{u}, \boldsymbol{v})) \\ \boldsymbol{h}(\boldsymbol{x}(\boldsymbol{u}, \boldsymbol{v})) \end{bmatrix}$. Next, we note that: $\nabla^2 \varphi(\boldsymbol{u}, \boldsymbol{v}) = \nabla \boldsymbol{x}(\boldsymbol{u}, \boldsymbol{v}) \begin{bmatrix} \boldsymbol{\nabla g}^T \\ \boldsymbol{\nabla h}^T \end{bmatrix}$, where $\nabla \boldsymbol{x}(\boldsymbol{u}, \boldsymbol{v})$ may be found from differentiating $\nabla_x \varphi(\boldsymbol{u}, \boldsymbol{v}) = \boldsymbol{0}$ as: $\nabla\big(\nabla_x \varphi(\boldsymbol{u}, \boldsymbol{v})\big) = \begin{bmatrix} \boldsymbol{\nabla g} \\ \boldsymbol{\nabla h} \end{bmatrix} + \nabla \boldsymbol{x}(\boldsymbol{u}, \boldsymbol{v}) \nabla_{xx} \varphi(\boldsymbol{u}, \boldsymbol{v}) = \boldsymbol{0}$.

Therefore, $\nabla \boldsymbol{x}(\boldsymbol{u}, \boldsymbol{v}) = \begin{bmatrix} \boldsymbol{\nabla g} \\ \boldsymbol{\nabla h} \end{bmatrix} [\nabla^2_{xx} \varphi(\boldsymbol{u}, \boldsymbol{v})]^{-1}$, and we finally get:

$$\nabla^2 \varphi(\boldsymbol{u}, \boldsymbol{v}) = \boldsymbol{\nabla g}\, (\nabla^2_{xx} \mathcal{L})^{-1}\, \boldsymbol{\nabla g}^T + \boldsymbol{\nabla h}\, (\nabla^2_{xx} \mathcal{L})^{-1}\, \boldsymbol{\nabla h}^T \tag{4.29}$$

An example of the dual function is given in the next section under convex optimization problems.

4.6.2 Duality in Convex Optimization Problems

In the case of convex optimization problems, if $\boldsymbol{x}^*$ is a regular point that solves the primal problem, and if $\boldsymbol{u}^*, \boldsymbol{v}^*$ are the associated Lagrange multipliers, then $(\boldsymbol{x}^*, \boldsymbol{u}^*, \boldsymbol{v}^*)$ is dual feasible and solves the dual problem. To develop these concepts, we consider the following quadratic programming (QP) problem:

$$\begin{aligned} &\text{Minimize } q(\boldsymbol{x}) = \tfrac{1}{2} \boldsymbol{x}^T \boldsymbol{Q} \boldsymbol{x} + \boldsymbol{c}^T \boldsymbol{x} \\ &\text{Subject to: } \boldsymbol{A}\boldsymbol{x} \geq \boldsymbol{b} \end{aligned} \tag{4.30}$$

where $\boldsymbol{Q}$ is symmetric and positive definite. Then, using Lagrange multiplier vector λ for the inequality constraint, a Lagrangian function for the QP problem is given as:

$$\mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}) = \frac{1}{2} \boldsymbol{x}^T \boldsymbol{Q} \boldsymbol{x} + \boldsymbol{c}^T \boldsymbol{x} - \boldsymbol{\lambda}^T (\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}) \tag{4.31}$$

The associated dual QP problem is defined as:

$$\begin{aligned} &\max_{\boldsymbol{x}, \boldsymbol{\lambda} \geq \boldsymbol{0}} \mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}) = \frac{1}{2} \boldsymbol{x}^T \boldsymbol{Q} \boldsymbol{x} + \boldsymbol{c}^T \boldsymbol{x} - \boldsymbol{\lambda}^T (\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}) \\ &\text{Subject to } \boldsymbol{Q}\boldsymbol{x} + \boldsymbol{c} - \boldsymbol{A}^T \boldsymbol{\lambda} = \boldsymbol{0} \end{aligned} \tag{4.32}$$

To obtain a solution, we first solve the constraint equation for $\boldsymbol{x}$ to get: $\boldsymbol{x}(\boldsymbol{\lambda}) = \boldsymbol{Q}^{-1}(\boldsymbol{A}^T \boldsymbol{\lambda} - \boldsymbol{c})$, and substitute it in the objective function to define the dual problem in terms of the dual function as:

$$\max_{\boldsymbol{\lambda} \geq \boldsymbol{0}} \varphi(\boldsymbol{\lambda}) = \frac{1}{2} \boldsymbol{\lambda}^T (\boldsymbol{A}\boldsymbol{Q}^{-1}\boldsymbol{A}^T) \boldsymbol{\lambda} + (\boldsymbol{A}\boldsymbol{Q}^{-1}\boldsymbol{c} + \boldsymbol{b})^T \boldsymbol{\lambda} - \frac{1}{2} \boldsymbol{c}^T \boldsymbol{Q}^{-1} \boldsymbol{c} \tag{4.33}$$

The gradient and Hessian of the dual function with respect to λ are computed as:

$$\nabla \varphi(\boldsymbol{\lambda}) = (\boldsymbol{A}\boldsymbol{Q}^{-1}\boldsymbol{A}^T) \boldsymbol{\lambda} + \boldsymbol{A}\boldsymbol{Q}^{-1}\boldsymbol{c} + \boldsymbol{b}, \qquad \nabla^2 \varphi(\boldsymbol{\lambda}) = \boldsymbol{A}\boldsymbol{Q}^{-1}\boldsymbol{A}^T \tag{4.34}$$

Download free eBooks at bookboon.com

If the optimal point is a regular point, then matrix $\boldsymbol{A}$ has full row rank. Then, from FONC, the solution to the dual QP problem is given as:

$$\boldsymbol{\lambda} = (\boldsymbol{A}\boldsymbol{Q}^{-1}\boldsymbol{A}^T)^{-1}(\boldsymbol{A}\boldsymbol{Q}^{-1}\boldsymbol{c} + \boldsymbol{b}) \tag{4.35}$$

where a non-negative solution $\boldsymbol{\lambda} \geq \boldsymbol{0}$ has been assumed.

As an example, we consider the following QP problem:

Example 4.6: quadratic optimization problem (Griva, Nash & Sofer, p.528)

Let the primal problem be defined as: $\min_x f(x) = x^2$ subject to $x \geq 1$. Then, a solution to the primal problem is given as: $x^* = 1$.

The Lagrangian function for the problem is formulated as: $\mathcal{L}(x, \lambda) = x^2 + \lambda(1 - x)$. The resulting dual optimization problem is defined as: $\max_{\lambda \geq 0} \mathcal{L}(x, \lambda) = x^2 + \lambda(1 - x)$ subject to $\nabla\mathcal{L}(x, \lambda) = 2x + \lambda = 0$.

Eliminating the constraint redefines the dual problem as: $\max_{\lambda \geq 0} \varphi(\lambda) = \lambda - \frac{\lambda^2}{4}$, with the solution: $\lambda^* = 2$.

Download free eBooks at bookboon.com

We may note that the saddle point condition is satisfied in this case, i.e.,

$$\max_{\lambda\geq 0} \mathcal{L}(x^*,\lambda) = \max_{\lambda\geq 0} \lambda - \frac{\lambda^2}{4} \leq \mathcal{L}(x^*,\lambda^*) = 1 \leq \min_{x\geq 1} \mathcal{L}(x,\lambda^*) = \min_{x\geq 1} x^2$$

with equality satisfied on both sides.

4.6.3 Local Duality Concepts

The existence of strong duality is only assured in the case of convex optimization problems. Nonetheless, using second order Taylor series expansion, any function can be locally approximated by a convex function. This prompts us to explore the possibility that the Lagrangian function is locally convex, and that strong duality may be locally achieved.

Towards this end, we consider the general optimization problem (4.15) with the Lagrangian function given by (4.17). Let $\boldsymbol{x}^\star$ denote a solution to the optimization problem; if $\boldsymbol{x}^\star$ is a regular point, then there exist unique Lagrange multipliers $(\boldsymbol{u}^*, \boldsymbol{v}^*)$, such that: $\nabla\mathcal{L}(\boldsymbol{x}^*, \boldsymbol{u}^*, \boldsymbol{v}^*) = 0$. As discussed in 4.6.1 above, a local dual function for the problem can be defined via (4.26), and the dual optimization problem can be locally defined via (4.17). The local problem can be solved via application of FONC. Let $(\boldsymbol{u}^*, \boldsymbol{v}^*)$ be the optimal solution to the dual problem, then $\mathcal{L}^*(\boldsymbol{u}^*, \boldsymbol{v}^*) = f(\boldsymbol{x}^*)$.

We note that local duality is applicable to both convex and non-convex problems. For example, we can apply local duality to the QP problem, to write:

$$\begin{aligned}
&\boldsymbol{x}(\boldsymbol{\lambda}) = \boldsymbol{Q}^{-1}(\boldsymbol{A}^T\boldsymbol{\lambda} - \boldsymbol{c}), \qquad \nabla\boldsymbol{g} = \boldsymbol{A}^T, \\
&\mathcal{L}^*(\boldsymbol{\lambda}) = \frac{1}{2}\boldsymbol{\lambda}^T(\boldsymbol{A}\boldsymbol{Q}^{-1}\boldsymbol{A}^T)\boldsymbol{\lambda} + (\boldsymbol{A}\boldsymbol{Q}^{-1}\boldsymbol{c} + \boldsymbol{b})^T\boldsymbol{\lambda} - \frac{1}{2}\boldsymbol{c}^T\boldsymbol{Q}^{-1}\boldsymbol{c}, \\
&\nabla\mathcal{L}^*(\boldsymbol{\lambda}) = -\boldsymbol{A}\boldsymbol{Q}^{-1}\boldsymbol{A}^T\boldsymbol{\lambda} + \boldsymbol{A}\boldsymbol{Q}^{-1}\boldsymbol{c} + \boldsymbol{b}, \text{ and } \nabla^2\mathcal{L}^*(\boldsymbol{\lambda}) = \boldsymbol{A}\boldsymbol{Q}^{-1}\boldsymbol{A}^T
\end{aligned} \tag{4.36}$$

As an example of a non-convex optimization problem we consider the following problem:

Example 4.7: Local duality

We consider the following optimization problem (Arora, p. 205):

$$\min_x f(x) = -x_1x_2, \text{ subject to } (x_1-3)^2 + x_2^2 = 5$$

The Lagrangian function and its derivative are given as:

$$\begin{aligned}
&\mathcal{L}(\boldsymbol{x},\boldsymbol{\lambda}) = -x_1x_2 + \lambda((x_1-3)^2 + x_2^2 - 5), \text{ where} \\
&\nabla_x\mathcal{L}(\boldsymbol{x},\boldsymbol{\lambda}) = \begin{bmatrix} -x_2 + 2\lambda(x_1-3) \\ -x_1 + 2\lambda x_2 \end{bmatrix} = \begin{bmatrix} 2\lambda & -1 \\ -1 & 2\lambda \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} - \begin{bmatrix} 6\lambda \\ 0 \end{bmatrix}
\end{aligned}$$

Download free eBooks at bookboon.com

Solving the FONC together with the equality constraint, the optimum solution to the problem is given as:

$$\boldsymbol{x}^* = (4,2), \lambda^* = 1, f^* = -8.$$

The Hessian at the optimum point is computed as: $\nabla^2(\boldsymbol{x}^*, \lambda^*) = \begin{bmatrix} 2 & -1 \\ -1 & 2 \end{bmatrix}$, and is positive definite. Therefore, using local duality theory, $\nabla_x \mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}) = 0$ is solved for x, which gives: $\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \frac{1}{4\lambda^2 - 1}\begin{bmatrix} 12\lambda^2 \\ 6\lambda \end{bmatrix}$. Next, the dual problem defined by $\max_{\boldsymbol{\lambda \geq 0}} \mathcal{L}(\boldsymbol{x}(\lambda), \lambda) = \frac{4(\lambda + \lambda^3 - 20\lambda^5)}{(4\lambda^2 - 1)^2}, \lambda \neq \pm\frac{1}{2}$, which has a local solution at $\lambda^* = 1$ with $\mathcal{L}(\boldsymbol{x}(\lambda^*), \lambda^*) = -8,$ which matches the primal solution.

4.6.4 Separable Problems

Dual methods are particularly powerful when applied to separable problems that are structured as:

$$\begin{aligned} &\min_{\boldsymbol{x}} f(\boldsymbol{x}) = \sum_i f_i(x_i) \\ &\text{Subject to: } \textstyle\sum_i g_i(x_i) \leq 0,\ \sum_i h_i(x_i) = 0 \end{aligned} \tag{4.37}$$

The dual function for the separable problem is formulated as:

$$\varphi(\boldsymbol{u}, \boldsymbol{v}) = \min_{\boldsymbol{x}} \left(\sum_i f_i(x_i) + u \sum_i g_i(x_i) + v \sum_i h_i(x_i) \right)$$

which decomposes into *m* separate single-variable problems given as: $\min_{x_i} f_i(x_i) + u g_i(x_i) + v h_i(x_i)$, which can be relatively easy to solve. Thus, the formulation of a dual problem becomes simple.

The next example shows how local duality can be applied to engineering problems that are separable.

Example 4.8: truss design problem (Belegundu and Chandrupatla, p. 274)

A truss contains a total of 16 elements of length $L_i = 30$ in, $i = 1, \ldots, 12$; $L_i = 30\sqrt{2}$ in, $i = 12, \ldots, 16$ and cross-sectional area x_i (design variable). The truss bears a load $P = 25{,}000$ lb at the tip. The weight of the structure is to be minimized with a bound on tip deflection, $\delta \leq 1$ in. The problem is formulated as:

$$\begin{aligned} &\min_{\boldsymbol{x}} \sum_{i=1}^{16} L_i x_i \\ &\text{Subject to: } \textstyle\sum_{i=1}^{16} \left(\frac{c_i}{x_i} - \alpha \right) \leq 0, x_i \geq x_i^L \text{ where } c_i = \frac{P L_i f_i^2}{E \delta_U},\ \alpha = \frac{1}{16},\ x_i^L = 10^{-6} \text{ in.} \end{aligned}$$

Download free eBooks at bookboon.com

The dual function is defined as: $\varphi(\mu) = \min_{x_i \geq x_i^L} \sum_{i=1}^{16} L_i x_i + \mu \left(\frac{c_i}{x_i} - \alpha \right)$

which leads to individual problems of the form: $\min_{x_i \geq x_i^L} \psi = L_i x_i + \mu \left(\frac{c_i}{x_i} - \alpha \right)$.

Application of FONC gives: if and if $x_i^* = \sqrt{\frac{\mu c_i}{L_i}}$ if $c_i > 0$, and $x_i^* = x_i^L$ if $c_i = 0$.

The resulting dual maximization problem is defined as: $\max_\mu \varphi(\mu) = 2 \sum_{i=1}^{16} \sqrt{c_i L_i} \sqrt{\mu} - \mu + c$

where c is a constant. Application of FONC then gives: $\mu = \left(\sum_i \sqrt{c_i L_i} \right)^2$.

For the given data, a closed-form solution is obtained as: $\mu^* = 1358.2$, $f^* = 1358.2$ in^3,

$\boldsymbol{x} = [5.67\ 4.25\ 4.25\ 2.84\ 2.84\ 1.42\ 1.42\ 10^{-6}\ 1.06\ 1.06\ 1.06\ 10^{-6}\ 1.77\ 1.77\ 1.77\ 1.77]$ in.

Download free eBooks at bookboon.com

5 Linear Programming Methods

Linear programming (LP) problems form an important subclass of the optimization problems. The distinguishing feature of the LP problems is that the objective function and the constraints are linear functions of the optimization variables. LP problems occur in many real-life economic situations where profits are to be maximized or costs minimized with constraints on resources. Specialized procedures, such as the Simplex method, were developed to solve the LPP. The simplex method divides the variables into basic and nonbasic, the latter being zero, in order to develop a basic feasible solution (BFS). It then iteratively updates basic variables thus generating a series of BFS, each of which carries a lower objective function value than the previous. Each time, the reduced costs associated with nonbasic variables are inspected to check for optimality. An optimum is reached when all the reduced costs are non-negative.

Learning Objectives: The learning objectives in this chapter are:

1. Understand the general formulation of a linear programming (LP) problem
2. Learn the Simplex method to solve LP problems and its matrix/tableau implementation
3. Understand the fundamental duality properties associated with LP problems
4. Learn sensitivity analysis applied to the LP problems
5. Grasp the formulation of KKT conditions applied to linear and quadratic programming problems
6. Learn to formulate and solve the linear complementarity problem (LCP)

5.1 The Standard LP Problem

The general LP problem is described in terms of minimization (or maximization) of a scalar objective function of n variables, that are subject to m constraints. These constraints may be specified as EQ (equality constraints), GE (greater than or equal to inequalities), or LE (less than or equal to inequalities). The variables themselves may be unrestricted in range, specified to be non-negative, or upper and/or lower bounded.

Mathematically, the LP problem is expressed as:

$$\min_{x_j} (\text{or } \max_{x_j}) \; z = \sum_{j=1}^{n} c_j x_j$$

$$\text{Subject to: } \sum_{j=1}^{n} a_{ij} x_j (\leq, =, \geq) b_i, \quad i = 1,2, \dots, m$$

$$x_j \geq x_{jL} \text{ (for some } j), \; x_j \leq x_{jU} \text{ (for some } j), \; x_j \text{ free (for some } j)$$

Download free eBooks at bookboon.com

While the general LP problem may be specified in different ways, the standard LP problem refers to a problem involving minimization of a scalar cost function, subject to only equality constraints and with optimization variables restricted to take on non-negative values. The inequality constraints can be converted to equality by adding (subtracting) slack (surplus) variables to LE (GE) type constraints. Further, the original variables can be replaced by new variables, which take on non-negative values. The standard LP problem is defined as:

$$\min_{x_i} z = \sum_{j=1}^{n} c_j x_j \tag{5.1}$$

Subject to: $\sum_{j=1}^{n} a_{ij} x_j = b_i\,,\ x_j \geq 0;\ \ i = 1,2,\dots,m$

The standard LP problem thus has the following characteristics:

1. It involves minimization of a scalar cost function.
2. The variables can only take on non-negative values, i.e., $x_i \geq 0$.
3. The r.h.s. is assumed to be non-negative, i.e., $b_j \geq 0$.

Additionally,

1. The constraints are assumed to be linearly independent, which implies that $rank(\boldsymbol{A}) = m$.
2. The problem is assumed to be well-formulated, which implies that $\min_{\boldsymbol{x}} \boldsymbol{c}^T\boldsymbol{x} < \infty$.

In the vector-matrix format, the standard LP problem is expressed as:

$$\min_{\boldsymbol{x}} z = \boldsymbol{c}^T\boldsymbol{x} \tag{5.2}$$

subject to $\boldsymbol{Ax} = \boldsymbol{b},\ \boldsymbol{x} \geq \boldsymbol{0}$

where $\boldsymbol{A} \in \mathbb{R}^{m\times n};\ \boldsymbol{x}, \boldsymbol{c} \in \mathbb{R}^n, \boldsymbol{b} \in \mathbb{R}^m$.

When encountered, exceptions to the standard LP problem formulation are dealt as follows:

1. A maximization problem is changed to a minimization problem by taking negative of the cost function, i.e., $\max_{\boldsymbol{x}} \boldsymbol{c}^T\boldsymbol{x} \equiv \min_{\boldsymbol{x}}(-\boldsymbol{c}^T\boldsymbol{x})$.
2. Any constant terms in z can be dropped.
3. Any $x_i \in \mathbb{R}$ (unrestricted in sign) is replaced by $x_i = x_i^+ - x_i^-$ where $x_i^+, x_i^- \geq 0$.
4. The inequality constraints are converted to equality constraints by the addition of slack variables (to LE constraint) or subtraction of surplus variables (from GE constraint).
5. If $b_j < 0$, the constraint is first multiplied by –1, followed by the introduction of slack or surplus variables.

Download free eBooks at bookboon.com

5.2 The Basic Solution to the LP Problem

We first note that the feasible set defined by linear equalities (and inequalities) in an LP problem is convex. In addition, the cost function is linear, hence convex. Therefore, the LP problem represents a convex optimization problem and a single global minimum for the problem exists.

Further, due to only equality constraints present in the problem, the optimum solution, if it exists, lies on the boundary of the feasible region. This is also true in the case of inequality constraints, since if none of the constraints is active, the application of first order optimality conditions to the problem would result in $c_i = 0, i = 1, \dots, n$, i.e., there will be no optimal solution.

Algebraically, the LP problem represents a linear system of m equations in n unknowns. Accordingly,

a) If $m = n$, the solution may be obtained from the constraints only.
b) If $m > n$, some of the constraints may be redundant, or the system may be inconsistent.
c) If $m < n$, the LP problem has an optimum solution, and can be solved using methods described below.

Download free eBooks at bookboon.com

Next, we consider the $m < n$ case, and assume that matrix $\boldsymbol{A}$ has full row rank; then, we arbitrarily choose independent (nonbasic) variables, to solve for the remaining (m) dependent (basic) variables. Let the system be transformed into canonical form: $\boldsymbol{I}_{(m)}\boldsymbol{x}_{(m)} + \boldsymbol{Q}\boldsymbol{x}_{(n-m)} = \boldsymbol{b}$; then, the general solution includes $(n-m)$ independent variables: $\boldsymbol{x}_{(n-m)}$, and (m) dependent variables: $\boldsymbol{x}_{(m)} = \boldsymbol{b} - \boldsymbol{Q}\boldsymbol{x}_{(n-m)}$. A particular solution to the linear system can be obtained by setting: $\boldsymbol{x}_{(n-m)} = \boldsymbol{0}$, and obtaining: $\boldsymbol{x}_{(m)} = \boldsymbol{b}$.

A basic solution $\boldsymbol{x}$ to a standard LP problem satisfies two conditions:

1. $\boldsymbol{x}$ is a solution to $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$.
2. The columns of $\boldsymbol{A}$ corresponding to the nonzero components of $\boldsymbol{x}$ are linearly independent.

Since $\boldsymbol{A}$ can have at the most m independent columns, it implies that $\boldsymbol{A}$ has at the most m nonzero components. When $\boldsymbol{A}$ has a full row rank, a basic solution is obtained by choosing $n-m$ variables as zero. The resulting solution, $\boldsymbol{x}$, contains m basic variables, $\boldsymbol{x}_{\boldsymbol{B}}$, and $n-m$ nonbasic variables, $\boldsymbol{x}_{\boldsymbol{N}}$, the latter taking on zero values. The columns of $\boldsymbol{A}$ corresponding to $\boldsymbol{x}_{\boldsymbol{B}}$ are termed as the basis vectors.

The set $\mathcal{S} = \{\boldsymbol{x}: \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}, \boldsymbol{x} \geq \boldsymbol{0}\}$ represents the feasible region for the LP problem. We note that a basic solution, $\boldsymbol{x} \in \mathcal{S}$, that is in the feasible region is termed as a basic feasible solution (BFS). Further, the feasible region is a polytope (polygon in n dimensions), and each BFS represents an extreme point (a vertex) of the polytope.

Let $\boldsymbol{x}^T = [\boldsymbol{x}_B,\ \boldsymbol{x}_N]$ where the basic variables occupy leading positions; we accordingly partition the cost function coefficients as: $\boldsymbol{c}^T = [\boldsymbol{c}_B, \boldsymbol{c}_N]$, and represent the constraint matrix as: $\boldsymbol{A} = [\boldsymbol{B}, \boldsymbol{N}]$, where $\boldsymbol{B}$ is a $m \times m$ nonsingular matrix and $\boldsymbol{N}$ is a $m \times (n-m)$; then, the original LP problem is reformulated as:

$$\begin{aligned} &\min_{\boldsymbol{x}} z = \boldsymbol{c}_B^T\boldsymbol{x}_B + \boldsymbol{c}_N^T\boldsymbol{x}_N, \\ &\text{Subject to } \boldsymbol{A}\boldsymbol{x} = [\boldsymbol{B}\ \ \boldsymbol{N}]\begin{bmatrix}\boldsymbol{x}_B \\ \boldsymbol{x}_N\end{bmatrix} = \boldsymbol{b},\ \boldsymbol{x}_B \geq \boldsymbol{0},\ \boldsymbol{x}_N \geq \boldsymbol{0} \end{aligned} \tag{5.3}$$

Then, a BFS corresponding to basis $\boldsymbol{B}$ is represented as: $\boldsymbol{x}^T = [\boldsymbol{x}_B,\ \boldsymbol{0}]$, $\boldsymbol{x}_{\boldsymbol{B}} = \boldsymbol{B}^{-1}\boldsymbol{b} > \boldsymbol{0}$. Since, by assumption, $rank\ (\boldsymbol{A}) = m$, $\boldsymbol{B}$, can be selected from the various permutations of the columns of $\boldsymbol{A}$. Further, since each basic solution has exactly m non-zero components, the total number of basic solutions is finite, and is given as: $\binom{n}{m} = \frac{n!}{m!(n-m)!}$. The number of BFS is smaller than number of basic solutions and can be determined by comparing the objective function values at the various basic solutions.

The Basic Theorem of Linear Programming (e.g., Arora, p. 201) states that if there is a feasible solution to the LP problem, there is a BFS; and if there is an optimum feasible solution, there is an optimum BFS. The basic LP theorem implies that an optimal BFS must coincide with one of the vertices of the feasible region. This fact can be used to compare the objective function value at all BFSs, and find the optimum by comparison if the number of vertices is small. Finally, there can also be multiple optimums if an active constraint boundary is parallel to the level curves of the cost function.

Download free eBooks at bookboon.com

5.3 The Simplex Method

The simplex method iteratively solves the standard LP problem. It does so by starting from a known BFS and successively moving to an adjacent BFS that carries a lower objective function value. Each move involves replacing a single variable in the basis with a new variable, such that the objective function value decreases. The previously nonbasic variable entering the basis is termed as entering basic variable (EBV), and the one leaving it is termed as leaving basic variable (LBV). An optimum is reached when no neighboring BFS with a lower objective function value can be found.

5.3.1 The Simplex Algorithm

In order to mathematically formulate the simplex algorithm, let $\boldsymbol{x}^T = [\boldsymbol{x_B}, \boldsymbol{x_N}]$ represent a non-basic solution to the LP problem, and let the constraints be expressed as: $\boldsymbol{B}\boldsymbol{x}_B + \boldsymbol{N}\boldsymbol{x}_N = \boldsymbol{b}$. Then, we can solve for $\boldsymbol{x}_B$ as: $\boldsymbol{x}_B = \boldsymbol{B}^{-1}(\boldsymbol{b} - \boldsymbol{N}\boldsymbol{x}_N)$, and substitute it in the objective function to obtain:

$$z = \boldsymbol{c}_B^T\boldsymbol{B}^{-1}\boldsymbol{b} + (\boldsymbol{c}_N^T - \boldsymbol{c}_B^T\boldsymbol{B}^{-1}\boldsymbol{N})\boldsymbol{x}_N = \boldsymbol{y}^T\boldsymbol{b} + \hat{\boldsymbol{c}}_N^T\boldsymbol{x}_N = \hat{z} + \hat{\boldsymbol{c}}_N^T\boldsymbol{x}_N \tag{5.4}$$

where $\boldsymbol{y}^T = \boldsymbol{c}_B^T\boldsymbol{B}^{-1}$ defines a vector of simplex (Lagrange) multipliers, $\hat{\boldsymbol{c}}_N^T = \boldsymbol{c}_N^T - \boldsymbol{y}^T\boldsymbol{N}$ represents the reduced costs for the nonbasic variables (reduced costs are zero for the basic variables), and $\hat{z} = \boldsymbol{y}^T\boldsymbol{b}$ represents the objective function value corresponding to a basic solution, where $y_i > 0$ represents an active constraint.

The significance of the reduced costs is as follows: let $\hat{c}_j \in \hat{\boldsymbol{c}}_N^T$; then, we note that assigning a nonbasic variable x_j a nonzero value δ_j will change the objective function by $\hat{c}_j\delta_j$. Therefore, any $\hat{c}_j < 0$ has the potential to decrease the value of z, and the corresponding x_j may be selected as the EBV. It is customary to select the variable x_j with the lowest $\hat{c}_j$ as the EBV.

To select an LBV, we examine the update to the basic solution from the introduction of EBV, given as: $\boldsymbol{x}_B = \widehat{\boldsymbol{b}} - \widehat{\boldsymbol{A}}_q x_q$, where $\widehat{\boldsymbol{b}} = \boldsymbol{B}^{-1}\boldsymbol{b}$, $\widehat{\boldsymbol{A}}_q = \boldsymbol{B}^{-1}\boldsymbol{A}_q$, and $\boldsymbol{A}_q$ represents the qth column of $\boldsymbol{A}$. Then x_q, can be increased so long as $\boldsymbol{x}_B \geq \boldsymbol{0}$; element wise considerations require that: $\hat{b}_i - \hat{A}_{i,q}x_q > 0$. Therefore, the maximum value of x_q is $\bar{x}_q = \min_i \left\{\frac{\hat{b}_i}{\hat{A}_{i,q}} : \hat{A}_{i,q} > 0\right\}$, is , and the variable corresponding to the lowest ratio from this ratio test is picked as LBV.

The steps involved in the Simplex algorithm are summarized below.

The Simplex Algorithm (Griva, Nash & Sofer, p.131):

1. Initialize: Find an initial BFS to start the algorithm; accordingly, determine $\boldsymbol{x_B}$, $\boldsymbol{x_N}$, $\boldsymbol{B}$, $\boldsymbol{N}$, $\boldsymbol{y}^T = \boldsymbol{c}_B^T\boldsymbol{B}^{-1}$, $\hat{z} = \boldsymbol{y}^T\boldsymbol{b}$.

Download free eBooks at bookboon.com

2. Optimality test: Compute $\widehat{\boldsymbol{b}} = \boldsymbol{B}^{-1}\boldsymbol{b},\ \hat{\boldsymbol{c}}_N^T = \boldsymbol{c}_N^T - \boldsymbol{y}^T\boldsymbol{N}$. Then, evaluate $\hat{\boldsymbol{c}}_N^T$ associated with current nonbasic variables. If all $\hat{c}_j > 0$, the optimal has been reached. Otherwise, select a variable x_q with $\hat{c}_q < 0$ as EBV.
3. Ratio test: Compute $\widehat{\boldsymbol{A}}_q = \boldsymbol{B}^{-1}\boldsymbol{A}_q$. Determine: $\min_i\left\{\frac{\hat{b}_i}{\hat{A}_{i,q}} : \hat{A}_{i,q} > 0\right\} = \frac{\hat{b}_p}{\hat{A}_{p,q}}$. Set $\hat{A}_{p,q}$ Set as the pivot element.
4. Update: Assign $x_q \leftarrow \frac{\hat{b}_p}{\hat{A}_{p,q}},\ \boldsymbol{x}_B \leftarrow \widehat{\boldsymbol{b}} - \hat{A}_q x_q,\ \hat{z} \leftarrow \hat{z} + \hat{c}_q x_q$. Update $\boldsymbol{x_B},\ \boldsymbol{x_N},\ \boldsymbol{B},\ \boldsymbol{N}$.

The following example illustrates the application of simplex algorithm to LP problems.

Example 5.1: The Simplex algorithm

We consider the following LP problem:

$$\max_{x_1,x_2} z = 3x_1 + 2x_2$$
Subject to: $2x_1 + x_2 \le 12,\ 2x_1 + 3x_2 \le 16;\ \ x_1 \ge 0, x_2 \ge 0$

Download free eBooks at bookboon.com

The problem is first converted to standard LP form by changing the sign of the objective function and adding slack variables s_1, s_2 to the LE constraints. The resulting optimization variables, cost coefficients, and constraint coefficients are given below:

$$\boldsymbol{x}^T = [x_1 \quad x_2 \quad s_1 \quad s_2], \qquad \boldsymbol{c}^T = [-3 \ -2 \ \ 0 \ \ 0], \qquad \boldsymbol{A} = \begin{bmatrix} 2 & 1 & 1 & 0 \\ 2 & 3 & 0 & 1 \end{bmatrix}, \qquad \boldsymbol{b} = \begin{bmatrix} 12 \\ 16 \end{bmatrix}$$

The steps of the Simplex algorithm for the problem are shown below:

Step 1:

$$\boldsymbol{x}_B = \begin{bmatrix} s_1 \\ s_2 \end{bmatrix} = \begin{bmatrix} 12 \\ 16 \end{bmatrix},\ \boldsymbol{x}_N = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix},\ \boldsymbol{c}_B^T = [0,0],\ \boldsymbol{c}_N^T = [-3,-2],\ \boldsymbol{B} = \boldsymbol{I},\ \hat{\boldsymbol{b}} = \boldsymbol{b} = \begin{bmatrix} 12 \\ 16 \end{bmatrix},\ \hat{z} = 0,$$

$$\boldsymbol{x}_B = \begin{bmatrix} s_1 \\ s_2 \end{bmatrix} = \begin{bmatrix} 12 \\ 16 \end{bmatrix},\ \boldsymbol{x}_N = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix},\ \boldsymbol{c}_B^T = [0,0],\ \boldsymbol{c}_N^T = [-3,-2],\ \boldsymbol{B} = \boldsymbol{I},\ \hat{\boldsymbol{b}} = \boldsymbol{b} = \begin{bmatrix} 12 \\ 16 \end{bmatrix},\ \hat{z} = 0,$$

Update: $x_1 \leftarrow 6,\ \boldsymbol{x}_B \leftarrow \begin{bmatrix} 0 \\ 4 \end{bmatrix},\ \hat{z} \leftarrow -18$

Step 2:

$$\boldsymbol{x}_B = \begin{bmatrix} x_1 \\ s_2 \end{bmatrix} = \begin{bmatrix} 6 \\ 4 \end{bmatrix},\ \boldsymbol{x}_N = \begin{bmatrix} x_2 \\ s_1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix},\ \boldsymbol{c}_B^T = [-3,0],\ \boldsymbol{c}_N^T = [-2,0],\ \boldsymbol{B} = \begin{bmatrix} 2 & 0 \\ 2 & 1 \end{bmatrix},\ \boldsymbol{N} = \begin{bmatrix} 1 & 1 \\ 3 & 0 \end{bmatrix},\ \hat{\boldsymbol{b}} = \begin{bmatrix} 6 \\ 4 \end{bmatrix},$$
$$\boldsymbol{y}^T = [-3/2, 0],\ \hat{\boldsymbol{c}}_N^T = [-1/2, 3/2],\ x_q = x_2,\ \hat{A}_1 = \begin{bmatrix} 1/2 \\ 2 \end{bmatrix},\ \left\{ \frac{\hat{b}_i}{\hat{A}_{i,1}} : \hat{A}_{i,1} > 0 \right\} = \{12, 2\},\ \hat{A}_{p,q} = \hat{A}_{2,2}$$

Update: $x_2 \leftarrow 2,\ \boldsymbol{x}_B \leftarrow \begin{bmatrix} 5 \\ 0 \end{bmatrix},\ \hat{z} \leftarrow -19$

Step 3:

$$\boldsymbol{x}_B = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 5 \\ 2 \end{bmatrix},\ \boldsymbol{x}_N = \begin{bmatrix} s_1 \\ s_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix},\ \boldsymbol{c}_B^T = [-3,-2],\ \boldsymbol{c}_N^T = [0,0],\ \boldsymbol{B} = \begin{bmatrix} 2 & 1 \\ 2 & 3 \end{bmatrix},\ \boldsymbol{N} = \boldsymbol{I},$$
$$\boldsymbol{y}^T = [-5/4,\ -1/4],\ \hat{\boldsymbol{c}}_N^T = [5/4,\ 1/4]$$

Since all $\hat{c}_j > 0$, an optimal has been reached and $z_{opt} = -19$.

5.3.2 Tableau Implementation of the Simplex Algorithm

It is customary to use tableaus to capture the essential information about the LP problem. A tableau is an augmented matrix that includes the constraints matrix, the right-hand-side (rhs), and the coefficients of the cost function (represented in the last row). Each preceding row of the tableau represents a constraint equation, and each column represents a variable. The tableau method implements the simplex algorithm by iteratively computing the inverse of the basis ($\boldsymbol{B}$) matrix.

Download free eBooks at bookboon.com

We consider a standard linear program with *n* variables and *m* equality constraints, and assume that at the current iteration the vectors of basic and nonbasic variables are represented as x_B and x_N, respectively. Then, the original linear program corresponds to the following tableau:

Basic	$\boldsymbol{x_B}$	$\boldsymbol{x_N}$	Rhs
$\boldsymbol{x_B}$	$\boldsymbol{B}$	$\boldsymbol{N}$	$\boldsymbol{b}$
$\boldsymbol{-z}$	$\boldsymbol{c_B^T}$	$\boldsymbol{c_N^T}$	0

In the above, basic variables are identified in the left-most column, the next columns *m* pertain to basis vectors and the right-most column represents the rhs. By pre-multiplying the tableau with the matrix: $\begin{bmatrix} \boldsymbol{B^{-1}} & 0 \\ \boldsymbol{-y^T} & 1 \end{bmatrix}, \boldsymbol{y^T} = \boldsymbol{c_B^T B^{-1}}$, we obtain the tableau representation in the current basis:

Basic	$\boldsymbol{x_B}$	$\boldsymbol{x_N}$	Rhs
$\boldsymbol{x_B}$	$\boldsymbol{I}$	$\boldsymbol{B^{-1}N}$	$\boldsymbol{B^{-1}b}$
$\boldsymbol{-z}$	$\boldsymbol{0}$	$\boldsymbol{c_N^T - y^T N}$	$\boldsymbol{- y^T b}$

where $\boldsymbol{y^T}$ represents the vector of Lagrange multipliers, $\boldsymbol{c_N^T - y^T N}$ represents the reduced costs, and $\boldsymbol{y^T b}$ represents the current objective function value.

The simplex iteration begins with the optimality test: we inspect the reduced costs for the nonbasic variables (represented in the last row under nonbasic variables), and pick the variable with most negative reduced cost (or another variable with negative reduced cost) as EBV. Next, we use the ratio test to determine LBV from the current basis. Once the pivot element has been identified, Gaussian elimination steps (elementary row operations) are completed to reduce the EBV column to a unit vector, effectively replacing LBV with EBV in the current basis. This completes an iteration of the simplex method. The process is then repeated till all the reduced costs are non-negative, thus signifying that an optimum has been reached.

We note that only the original matrices: $\boldsymbol{A, b, c}$ and the inverse of current basis matrix ($\boldsymbol{B^{-1}}$) are needed to compute the remaining entries in the current tableau. Further, only the EBV column among the nonbasic variable columns needs to be computed. These facts are useful when developing a computationally efficient implementation of the simplex method. Additionally, some mechanism is needed to update the representation of the inverse matrix for the next iteration. Computationally efficient ways of doing so are discussed in (Griva, Nash & Sofer, Ch. 7).

Download free eBooks at bookboon.com

Some abnormal terminations of the Simplex algorithm are described as follows:

1. If the reduced cost for a nonbasic variable in the final tableau is zero, then there exists a possibility for multiple optimum solutions with equal cost function value. This happens when cost function contours (level curves) are parallel to one of the constraint boundaries.
2. If the reduced cost is negative but the pivot step cannot be completed due to all coefficients in the LBV column being negative, it reveals a situation where the cost function is unbounded below.
3. If, at some point during Simplex iterations, a basic variable attains a zero value (i.e., the rhs has a zero), it is called degenerate variable and the corresponding BFS is termed as degenerate solution, as the degenerate row hence forth becomes the pivot row, with no improvement in the objective function. While it is theoretically possible for the algorithm to fail by cycling between two degenerate BFSs, this is not known to happen in practice.

An example for tableau implementation of the simplex method is presented below.

Download free eBooks at bookboon.com

Example 5.2: the Tableau method

As an example of the tableau method, we resolve example 5.1 using tableaus, where the optimization problem is stated as:

$$\max_{x_1,x_2} z = 3x_1 + 2x_2$$
Subject to: $2x_1 + x_2 \leq 12,\ 2x_1 + 3x_2 \leq 16;\ \ x_1 \geq 0, x_2 \geq 0$

The problem is first converted to the standard LP form. The constraints and cost function coefficients were entered in an initial simplex tableau, where the EBV, LBV, and the pivot element are identified underneath the tableau:

Basic	$\boldsymbol{x_1}$	$\boldsymbol{x_2}$	$\boldsymbol{s_1}$	$\boldsymbol{s_2}$	Rhs
$\boldsymbol{s_1}$	2	1	1	0	12
$\boldsymbol{s_2}$	2	3	0	1	16
$\boldsymbol{z}$	-3	-2	0	0	0

EBV: x_1, LBV: s_1, pivot: (1,1)

The subsequent simplex iterations result in the series of tableaus appearing below:

Basic	$\boldsymbol{x_1}$	$\boldsymbol{x_2}$	$\boldsymbol{s_1}$	$\boldsymbol{s_2}$	rhs
$\boldsymbol{x_1}$	1	0.5	0.5	0	6
$\boldsymbol{s_2}$	0	2	-1	1	4
$-\boldsymbol{z}$	0	-0.5	1.5	0	18

EBV: x_2, LBV: s_2, pivot: (2,2)

Basic	$\boldsymbol{x_1}$	$\boldsymbol{x_2}$	$\boldsymbol{s_1}$	$\boldsymbol{s_2}$	rhs
$\boldsymbol{x_1}$	1	0	0.75	-0.25	5
$\boldsymbol{x_2}$	0	1	-0.5	0.5	2
$-\boldsymbol{z}$	0	0	1.25	0.25	19

At this point, since all reduced costs are positive, an optimum has been reached with:

$x_1^* = 5, x_2^* = 2, z_{opt} = -19.$

5.3.1 Final Tableau Properties

The final tableau from the simplex algorithm has certain fundamental properties that relate to the initial tableau. To reveal those properties, we consider the following optimization problem:

$$\begin{aligned} &\max_{\boldsymbol{x}}\ z = \boldsymbol{c}^T\boldsymbol{x} \\ &\text{Subject to: } \boldsymbol{Ax} \leq \boldsymbol{b},\ \boldsymbol{x} \geq \boldsymbol{0} \end{aligned} \tag{5.5}$$

Download free eBooks at bookboon.com

Adding surplus variables to the constraints results in the following standard LP problem:

$$\begin{aligned} &\min_{\boldsymbol{x}} \; z = -\boldsymbol{c}^T\boldsymbol{x} \\ &\text{Subject to: } \boldsymbol{A}\boldsymbol{x} + \boldsymbol{I}\boldsymbol{s} = \boldsymbol{b},\;\; \boldsymbol{x} \geq \boldsymbol{0} \end{aligned} \tag{5.6}$$

An initial tableau for this problem is given as:

Basic	$\boldsymbol{x}$	$\boldsymbol{s}$	Rhs
$\boldsymbol{s}$	$\boldsymbol{A}$	$\boldsymbol{I}$	$\boldsymbol{b}$
$-\boldsymbol{z}$	$-\boldsymbol{c}^T$	$\boldsymbol{0}$	0

Assuming that the same order of the variables is maintained, then at the termination of the Simplex algorithm the final tableau is given as:

Basic	$\boldsymbol{x}$	$\boldsymbol{s}$	rhs
$\boldsymbol{x_B}$	$\widetilde{\boldsymbol{A}}$	$\boldsymbol{S}$	$\widetilde{\boldsymbol{b}}$
$-\boldsymbol{z}$	$\hat{\boldsymbol{c}}^T$	$\boldsymbol{y}^T$	z^*

We note that the coefficients in the final tableau are related to those in the initial tableau as follows:

$$\widetilde{\boldsymbol{A}} = \boldsymbol{S}\boldsymbol{A},\;\; \widetilde{\boldsymbol{b}} = \boldsymbol{S}\boldsymbol{b}, \qquad \hat{\boldsymbol{c}}^T = \boldsymbol{y}^T\boldsymbol{A} - \boldsymbol{c}^T, \qquad z^* = \boldsymbol{y}^T\boldsymbol{b} \tag{5.7}$$

Thus, given the initial tableau $(\boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c}^T)$ and the final coefficients in the slack variable columns: $(\boldsymbol{y}^T, \boldsymbol{S})$, we can reconstruct the final tableau as:

$$[Tab]_{\text{final}} = \begin{bmatrix} \boldsymbol{S} & \boldsymbol{0} \\ \boldsymbol{y}^T & 1 \end{bmatrix} [Tab]_{\text{initial}} \tag{5.8}$$

Therefore, in a computer implementation of the Simplex algorithm, only the coefficients: $\boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c}^T, \boldsymbol{y}^T, \boldsymbol{S}$ need to be stored in order to recover the final tableau when the algorithm terminates.

5.3.2 Obtaining an Initial BFS

The starting point of the simplex algorithm is a valid BFS. This is trivial in the case of a maximization problems modeled with LE constraints (Example 5.1), where an obvious initial BFS is to choose the slack variables as the basic variables. Initial BFS is not so obvious when the problem involves GE or EQ constraints. It is so because the feasible region in the problem does not normally include the origin. Then, in order to initiate the simplex algorithm, we need to choose an initial $\boldsymbol{B}$ matrix, such that $\boldsymbol{B}\boldsymbol{x}_B = \boldsymbol{b}$ yields a non-negative solution for $\boldsymbol{x}_B$. The two-phase Simplex method described below obtains an initial BFS by first solving an auxiliary LP problem.

Download free eBooks at bookboon.com

The two-phase Simplex method works as follows: we add a set of m auxiliary variables $\tilde{\boldsymbol{x}}$, to the original optimization variables $\boldsymbol{x}$, and define an auxiliary LP problem where the auxiliary objective function is selected to reduce the auxiliary variables. The auxiliary problem is defined as:

$$\begin{aligned} &\min_{\tilde{x}_i} \sum_{i=1}^{m} \tilde{x}_i \\ &\text{Subject to: } [\boldsymbol{A} \quad \boldsymbol{I}] \begin{bmatrix} \boldsymbol{x} \\ \tilde{\boldsymbol{x}} \end{bmatrix} = \boldsymbol{b}, \boldsymbol{x} \geq \boldsymbol{0}, \tilde{\boldsymbol{x}} \geq \boldsymbol{0} \end{aligned} \tag{5.9}$$

The auxiliary problem is solved in Phase I of the Simplex algorithm. We note that $\tilde{\boldsymbol{x}} = \boldsymbol{b}$ is a valid BFS for the auxiliary problem and serves as a starting point for Phase I Simplex algorithm. Further, since only the GE and EQ constraints require auxiliary variables, their number can be accordingly chosen less than or equal to m.

The starting tableau for the Phase I Simplex algorithm is given as:

Basic	$\boldsymbol{x_B}$	$\boldsymbol{x_N}$	$\tilde{\boldsymbol{x}}$	Rhs
$\boldsymbol{x_B}$	$\boldsymbol{B}$	$\boldsymbol{N}$	$\boldsymbol{I}$	$\boldsymbol{b}$
$-\boldsymbol{z}$	$\boldsymbol{c}_B^T$	$\boldsymbol{c}_N^T$	$\boldsymbol{0}$	$\boldsymbol{0}$
$-\boldsymbol{z_a}$	$\boldsymbol{0}$	$\boldsymbol{0}$	$\boldsymbol{1}^T$	$\boldsymbol{0}$

Download free eBooks at bookboon.com

where $\mathbf{1}^T = [1, \ldots, 1]$ represents a unit vector. The first step in the Phase I Simplex is to make auxiliary variables the basic variables. This is done by row reductions aimed to generate unit vectors in the basis columns, which results in the following tableau:

Basic	$\boldsymbol{x_B}$	$\boldsymbol{x_N}$	$\widetilde{\boldsymbol{x}}$	Rhs
$\boldsymbol{x_B}$	$\boldsymbol{B}$	$\boldsymbol{N}$	$\boldsymbol{I}$	$\boldsymbol{b}$
$-\boldsymbol{z}$	$\boldsymbol{c_B^T}$	$\boldsymbol{c_N^T}$	$\mathbf{0}$	0
$-\boldsymbol{z_a}$	$-\mathbf{1}^T\boldsymbol{B}$	$-\mathbf{1}^T\boldsymbol{N}$	$\mathbf{0}$	$-\mathbf{1}^T\boldsymbol{b}$

The Phase I Simplex algorithm continues till all the reduced costs in the auxiliary objective row become non-negative and the auxiliary objective function value reduces to zero, thus signaling the end of Phase I. If the auxiliary objective value at the end of Phase I does not equal zero, it indicates that no feasible solution to the original problem exists.

Once the auxiliary problem has been solved, we turn to the original problem, and drop the auxiliary objective (z_a) row and the auxiliary variable $(\widetilde{\boldsymbol{x}})$ columns from the current tableau (or ignore them). We then follow up with further Simplex iterations in Phase II of the algorithm till an optimum value for z is obtained.

Two examples involving GE and EQ constraints are solved below to illustrate the implementation of the two-phase Simplex algorithm.

Example 5.3: Two-phase Simplex algorithm for GE constraints

We consider the following LP problem:

$$\max_{x_1,x_2} z = 3x_1 + 2x_2$$
Subject to: $3x_1 + 2x_2 \geq 12,\ 2x_1 + 3x_2 \leq 16,\ x_1 \geq 0, x_2 \geq 0$

We first convert the problem to standard form by subtracting surplus variable (s_1) from GE constraint and adding slack variable (s_2) to LE constraint. The standard form LP problem is given as:

$$\min_{x_1,x_2} z = -3x_1 - 2x_2$$
Subject to: $3x_1 + 2x_2 - \mathrm{s}_1 = 12,\ 2x_1 + 3x_2 + \mathrm{s}_2 = 16;\ x_1, x_2, s_1, s_2 \geq 0$

There is no obvious BFS to start the simplex algorithm. To solve the problem using two-phase simplex method, we add an auxiliary variable a_1 to GE constraint and define the following auxiliary LP problem:

$$\min_{x_1,x_2} z_a = a_1$$
Subject to: $3x_1 + 2x_2 - s_1 + a_1 = 12,\ 2x_1 + 3x_2 + s_2 = 16;\ x_1, x_2, s_1, s_2, a_1 \geq 0$

Download free eBooks at bookboon.com

The starting tableau for the auxiliary problem (Phase I) is given as:

Basic	x_1	x_2	s_1	s_2	a_1	Rhs
	3	2	-1	0	1	12
s_2	2	3	0	1	0	16
$-z$	-3	-2	0	0	0	0
$-z_a$	0	0	0	0	1	0

We first a_1 bring into the bases by reducing the a_1 column to a unit vector.

Basic	x_1	x_2	s_1	s_2	a_1	Rhs
s_1	3	2	-1	0	1	12
s_2	2	3	0	1	0	16
$-z$	-3	-2	0	0	0	0
$-z_a$	-3	-2	1	0	0	-12

EBV: x_1, LBV: s_1, pivot: (1,1)

The above step is followed by an additional Simplex iteration to reach the end of Phase I. The resulting final tableau for phase I is shown below:

Basic	x_1	x_2	s_1	s_2	a_1	Rhs
x_1	1	2/3	-1/3	0	1/3	4
s_2	0	5/3	2/3	1	-2/3	8
$-z$	0	0	-1	0	1	12
$-z_a$	0	0	0	0	1	0

Since the auxiliary variable is now nonbasic and the auxiliary objective has a zero value, the auxiliary problem has been solved. To turn to the original problem, we drop the z_a row and the a_1 column from the tableau. This results in the tableau below, which represents a valid BFS: $x_1 = 4, s_2 = 8$ to start Phase II.

Basic	x_1	x_2	s_1	s_2	Rhs
x_1	1	2/3	-1/3	0	4
s_2	0	5/3	2/3	1	8
$-z$	0	0	-1	0	12

EBV: s_1, LBV: s_2, pivot: (2,3)

Download free eBooks at bookboon.com

Phase II: To continue, we perform another iteration of the Simplex algorithm leading to the final tableau:

Basic	$\boldsymbol{x_1}$	$\boldsymbol{x_2}$	$\boldsymbol{s_1}$	$\boldsymbol{s_2}$	Rhs
$\boldsymbol{x_1}$	1	3/2	0	1/2	8
$\boldsymbol{s_1}$	0	5/2	1	3/2	12
$-\boldsymbol{z}$	0	5/2	0	3/2	24

At this point the original LP problem has been solved and the optimal solution is given as:

$x_1^* = 8,\ x_2^* = 0,\ z^* = -24.$

The second example of the two-phase simplex method involves equality constraints and bounds on the optimization variables.

Example 5.4: Two-phase Simplex algorithm for EQ constraints

We consider the following LP problem:

$\min\limits_{x_1,x_2}\ z = 2x_1 + x_2$
Subject to: $x_1 + x_2 = 3,\ 0 \le x_1 \le 2,\ 0 \le x_2 \le 2$

Download free eBooks at bookboon.com

We first add slack variables s_1, s_2 to the LE constraints. The resulting standard LP problem is given as:

$$\min_{x_1,x_2} z = 2x_1 + x_2$$
Subject to: $x_1 + x_2 = 3,\ x_1 + s_1 = 2,\ x_2 + s_2 = 2;\ x_1, x_2, s_1, s_2 \geq 0$

We note that no obvious BFS for the problem exists. In order to solve the problem via two-phase simplex method, we add an auxiliary variable a_1 to the EQ constraint and define the following auxiliary problem:

$$\min_{x_1,x_2} z_a = a_1$$
Subject to: $x_1 + x_2 + a_1 = 3,\ x_1 + s_1 = 2,\ x_2 + s_2 = 2;\ x_1, x_2, s_1, s_2, a_1 \geq 0$

The starting tableau for the auxiliary problem is given below:

Basic	x_1	x_2	s_1	s_2	a_1	Rhs
	1	1	0	0	1	3
s_1	1	0	1	0	0	2
s_2	0	1	0	1	0	2
$-z$	2	1	0	0	0	0
$-z_a$	0	0	0	0	1	0

First, the auxiliary variable is made basic by producing a unit vector in the column. This is followed by additional Simplex iterations to reach the Phase I solution as shown in the tableaus below:

Basic	x_1	x_2	s_1	s_2	a_1	Rhs
a_1	1	1	0	0	1	3
s_1	1	0	1	0	0	2
s_2	0	1	0	1	0	2
$-z$	2	1	0	0	0	0
$-z_a$	-1	-1	0	0	0	-3

EBV: x_1, LBV: s_1, pivot: (2,1)

Basic	x_1	x_2	s_1	s_2	a_1	Rhs
a_1	0	1	-1	0	1	1
x_1	1	0	1	0	0	2
s_2	0	1	0	1	0	2
$-z$	0	1	2	0	0	-4
$-z_a$	0	-1	1	0	0	-1

EBV: x_2, LBV: a_1, pivot: (1,2)

Download free eBooks at bookboon.com

Basic	x_1	x_2	s_1	s_2	a_1	Rhs
x_2	0	1	-1	0	1	1
x_1	1	0	1	0	0	2
s_2	0	0	1	1	-1	1
$-z$	0	0	-1	0	-1	-5
$-z_a$	0	0	0	0	1	0

At this point, since the reduced costs are non-negative and the auxiliary objective has a zero value, Phase I Simplex is completed with the initial BFS: $x_1 = 2, x_2 = 1$. After dropping the auxiliary variable column and the auxiliary objective row, the starting tableau for Phase II Simplex is given as:

Basic	x_1	x_2	s_1	s_2	Rhs
x_2	0	1	-1	0	1
x_1	1	0	1	0	2
s_2	0	0	1	1	1
$-z$	0	0	-1	0	-5

We follow the initial step with further simplex iterations. The optimum is reached in one iteration and the final tableau is given as:

Basic	x_1	x_2	s_1	s_2	Rhs
x_2	0	1	0	1	2
x_1	1	0	0	-1	1
s_2	0	0	1	1	1
$-z$	0	0	0	1	-4

Since the reduced costs are non-negative, the current solution is optimal, i.e., $x_1^* = 1, x_2^* = 2, z^* = 4$. Later, we show that this problem is more easily solved via dual Simplex method (Sec. 5.4.2).

5.4 Postoptimality Analysis

Postoptimality, or sensitivity analysis, aims to study how variations in the original problem parameters affect the optimum solution. Postoptimality analysis serves the following purposes:

1. To help in managerial decision making, regarding the potential effects of increase/decrease in resources or raising/lowering the prices.
2. To analyze the effect of modeling errors, reflected in the uncertainty in parameter values in the coefficient matrices $(\boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c}^T)$ on the final LP solution.

Download free eBooks at bookboon.com

In postoptimality analysis, we are interested to explore the effects of parametric changes in $b_i, c_j,$ and A_{ij} on the optimal solution. There are five basic parametric changes affecting the LP solution (Arora, p. 229):

1. Changes in cost function coefficients, c_j; these changes affect the level curves of the function.
2. Changes in resource limitations, b_i; these changes affect the set of active constraints.
3. Changes in constraint coefficients, a_{ij}; these changes affects the active constraint gradients.
4. The effect of including additional constraints
5. The effect of including additional variables

We note that we can use the final tableau to study the effects of parameter changes on the optimal solution. Toward that end, we recall, from Sec. 5.3.1, that the instantaneous cost function value in the Simplex algorithm is represented as: $z = \boldsymbol{y}^T\boldsymbol{b} + \hat{\boldsymbol{c}}_N^T\boldsymbol{x}_N$, where $\boldsymbol{y}^T = \boldsymbol{c}_B^T\boldsymbol{B}^{-1}$ and $\hat{\boldsymbol{c}}_N^T = \boldsymbol{c}_N^T - \boldsymbol{y}^T\boldsymbol{N}$.

Download free eBooks at bookboon.com

Then, writing $z = \sum_i y_i b_i + \sum_j \hat{c}_j x_j$, and taking the differentials with respect to b_i, c_j, we obtain: $\delta\hat{c}_j = \delta c_j - \boldsymbol{y}^T \delta \boldsymbol{N}_j$, $\delta z = \sum_i y_i \delta b_i + \sum_j \delta\hat{c}_j x_j$, where $\boldsymbol{N}_j$ represents the jth column of $\boldsymbol{N}$. The above formulation may be used to analyze the effects of changes to b_i, c_j, and $\boldsymbol{N}_j$ on z. Those results are summarized below (Belegundu & Chandrupatla, p. 167):

1. **Changes to the resource constraints (rhs).** A change in b_i has the effect of moving the associated constraint boundary. Then,
   a) If the constraint is currently active $(y_i > 0)$, the change will affect the current basic solution: $x_{\boldsymbol{B}} = \tilde{\boldsymbol{b}}$, as well as z_{opt}. If the new $x_{\boldsymbol{B}}$ is feasible, then $z_{opt} = \boldsymbol{y}^T \boldsymbol{b}$ is the new optimum value. If the new $x_{\boldsymbol{B}}$ is infeasible, then dual Simplex steps may be used to restore feasibility (and hence optimality).
   b) If the constraint is currently non-active $(y_i = 0)$, then z_{opt} and $x_{\boldsymbol{B}}$ are not affected.

2. **Changes to the objective function coefficients.** Changes to c_j affect the level curves of z. Then,
   a) If $c_j \in \boldsymbol{c}_B$, then since the new $\hat{c}_j \neq 0$, Gauss-Jordan eliminations are needed to return x_j to the basis. If optimality is lost in the process (any $\hat{c}_j < 0$), further Simplex steps will be needed to restore optimality. If optimality is not affected, then $z_{opt} = \boldsymbol{y}^T \boldsymbol{b}$ is the new optimum.
   b) If $c_j \in \boldsymbol{c}_N$, though it does not affect z, still $\hat{c}_j$ needs to be recomputed and checked for optimality.

3. **Changes to the coefficient matrix.** Changes to the coefficient matrix affect the constraint boundaries. For a change in $\boldsymbol{A}_j$ (jth column of $\boldsymbol{A}$),
   a) If $\boldsymbol{A}_j \in \boldsymbol{B}$, corresponding to a basic variable, then Gauss-Jordan eliminations are needed to reduce A_j to a unit vector; then $\hat{c}_j$ needs to be recomputed and checked for optimality.
   b) If $\boldsymbol{A}_j \in \boldsymbol{N}$, corresponding to a nonbasic variable, then the reduced cost $\hat{c}_j$ needs to be recomputed and checked for optimality.

4. **Adding Variables.** If we add a new variable x_{n+1} to the problem, then the cost function is updated as: $z = \boldsymbol{c}^T \boldsymbol{x} + c_{n+1} x_{n+1}$. In addition, a new column A_{n+1} is added to the constraint matrix. The reduced cost corresponding to the new column is computed as: $c_{n+1} - \boldsymbol{y}^T A_{n+1}$. Then, if this cost is positive, optimality is maintained; otherwise, further Simplex iterations are needed to recover optimality.

Download free eBooks at bookboon.com

5. **Adding inequality Constraints.** Assume that we add an inequality constraint to the problem. Adding a constraint adds a row and the associated slack/surplus variable adds a column to the tableau. In this case, we need to check if adding a column to the basis affects the current optimum. We define an augmented $\boldsymbol{B}$ matrix as: $\boldsymbol{B} = \begin{bmatrix} \boldsymbol{B} & 0 \\ \boldsymbol{a}_B^T & 1 \end{bmatrix}$, where $\boldsymbol{B}^{-1} = \begin{bmatrix} \boldsymbol{B}^{-1} & 0 \\ \boldsymbol{a}_B^T\boldsymbol{B}^{-1} & 1 \end{bmatrix}$, and write the augmented final tableau as:

Basic	$\boldsymbol{x_B}$	$\boldsymbol{x_N}$	Rhs
$\boldsymbol{x_B}$	$\boldsymbol{I}$	$\boldsymbol{B^{-1}N}$	$\boldsymbol{B^{-1}b}$
$\boldsymbol{x_{n+1}}$	$\boldsymbol{I}$	$\boldsymbol{a_B^TB^{-1}N}$	$\boldsymbol{a_B^TB^{-1}b} + b_{n+1}$
$\boldsymbol{-z}$	$\boldsymbol{0}$	$\boldsymbol{c_N^T - y^TN}$	$\boldsymbol{-y^Tb}$

Since the reduced costs are not affected, if $\boldsymbol{a_B^TB^{-1}b} + b_{n+1} > 0$, optimality is maintained. If not, we choose this row as the pivot row and apply dual Simplex steps (Sec. 5.5.2) to recover optimality.

Further results on sensitivity analysis involve parametric linear programming, aimed at analyzing the range of parameters values in the perturbed solution that maintain feasibility and/or optimality. These ranges are normally reported by commercial analysis software. Interested readers may consult Sec. 6.5 in (Griva, Nash & Sofer, 2009).

The following problem adopted from (Belegundu & Chandrupatla, p. 122) is used to illustrate the ideas presented in this section.

Example 5.5: Postoptimality Analysis

A vegetable farmer has the choice to grow tomatoes, green peppers, or cucumbers on his 200 acre farm. The man-days/per acre needed for growing the three vegetables are 6, 7 and 5, respectively. A total of 500 man-hours are available. The yield/acre for the three vegetables are in the ratios: 4.5:3.6:4.0. We wish to determine the optimum crop combination that maximizes total yield.

The optimization problem was solved using the Simplex method. The initial and the final tableaus for the problem are reproduced below:

Initial:

Basic	$\boldsymbol{x_1}$	$\boldsymbol{x_2}$	$\boldsymbol{x_3}$	$\boldsymbol{s_1}$	$\boldsymbol{s_2}$	Rhs
$\boldsymbol{s_1}$	1	1	1	1	0	200
$\boldsymbol{s_2}$	6	7	5	0	1	500
$\boldsymbol{-z}$	-4.5	-3.6	-4	0	0	0

Download free eBooks at bookboon.com

Final:

Basic	x_1	x_2	x_3	s_1	s_2	Rhs
s_1	-0.2	-0.4	0	1	-0.2	100
x_3	1.2	1.4	1	0	0.2	100
$-z$	0.3	2	0	0	0.8	400

From the final tableau, the optimum crop combination is given as: $x_1^* = 0, x_2^* = 0, x_3^* = 100$, with $z^* = 400$. Further, the shadow prices for the slack variables are: $\boldsymbol{y}^T = [0,\ 0.8]$, with $z^* = \boldsymbol{y}^T\boldsymbol{b} = 400$.

Next, without re-solving the problem, we wish to answer the following questions:

a) If an additional 50 acres are added, what is the expected change in yield? The answer is found from: $z^* = \boldsymbol{y}^T(\boldsymbol{b} + \Delta)$ where $\Delta = [50,0]^T$, with $z^* = 400$, i.e., there is no expected change in yield. This also means that the land area constraint is not binding in the current optimum solution.

b) If an additional 50 man-days are added, what is the expected change in yield? The answer is found from: $z^* = \boldsymbol{y}^T(\boldsymbol{b} + \Delta)$ where $\Delta = [0, 50]^T$, with $z^* = 440$ i.e., the yield increases by 40 units. This also means that the man-days constraint is binding in the optimum solution.

Download free eBooks at bookboon.com

c) If the yield/acre for tomatoes increases by 10%, how is the optimum affected? The answer is found by re-computing the reduced costs as: $\hat{\boldsymbol{c}}^T = \boldsymbol{y}^T\boldsymbol{A} - \boldsymbol{c}^T = [-0.15, 2, 0]$. Since a reduced cost is now negative, additional Simplex steps are needed to regain optimality. This is done and the new optimum is: $x_1^* = 83.33, x_2^* = 0, x_3^* = 0$ with $z^* = 412.5$.

d) If the yield/acre for cucumbers drops by 10%, how is the optimum be affected? The answer is found by re-computing the reduced costs as: $\hat{\boldsymbol{c}}^T = \boldsymbol{y}^T\boldsymbol{A} - \boldsymbol{c}^T = [0.3, 2, 0.4]$. The reduced costs are non-negative, but $\boldsymbol{x_3}$ is no more a basic variable. Regaining the basis results in reduced cost for $\boldsymbol{x_1}$ becoming negative. Additional Simplex steps are performed to regain optimality, and the new optimum is: $x_1^* = 83.33, x_2^* = 0, x_3^* = 0$ with $z^* = 375$.

e) If the man-hours needed to grow green peppers increase to 5/acre, how is the optimum affected? The answer is found by re-computing the reduced cost: $\hat{c}_2 = y^T A_2 - c_2 = 0.4$. Since x_2 was non-basic and the revised reduced cost is non-negative, there is no change in the optimum solution.

5.5 Duality Theory for the LP Problems

In this section we extend the Lagrangian duality (Sec. 4.5) to the LP problems. Duality theory applies to practical LP problems in engineering and economics. In engineering, for example, the primal problem in electric circuit theory may be posed in terms of electric potential, and its dual in terms of current flow. Similarly, an optimization problem in mechanics may be modeled with strains, and its dual modeled with stresses. In economics, if the primal problem seeks to optimize price per unit of product, its dual may seek to minimize cost per unit of resources.

The LP duality is defined in the following terms: associated with every LP problem is a dual problem that is formulated in terms of dual variables, i.e., the Lagrange multipliers. In the symmetric form of duality, the primal (P) and the dual (D) LP problems are stated as:

$$\begin{aligned} &\text{(P)} \ \max_{\boldsymbol{x}} \ z = \boldsymbol{c}^T\boldsymbol{x}, \ \text{ subject to } \boldsymbol{A}\boldsymbol{x} \le \boldsymbol{b}, \ \boldsymbol{x} \ge \boldsymbol{0} \\ &\text{(D)} \ \min_{\boldsymbol{y}} \ w = \boldsymbol{y}^T\boldsymbol{b}, \ \text{ subject to } \boldsymbol{y}^T\boldsymbol{A} \ge \boldsymbol{c}^T, \ \boldsymbol{y} \ge \boldsymbol{0} \end{aligned} \tag{5.10}$$

where $\boldsymbol{x} \in \mathbb{R}^n$ denotes the primal variables and $\boldsymbol{y} \in \mathbb{R}^m$ denotes the dual variables. We note that, based on the definition of duality, the dual of the dual (D) is the same as primal (P).

To define the dual of an arbitrary LP problem we first convert the problem into an equivalent problem of the primal form. The dual problem can then be formulated accordingly. For example, when (P) is given in the standard LP form, then (D) takes the following form:

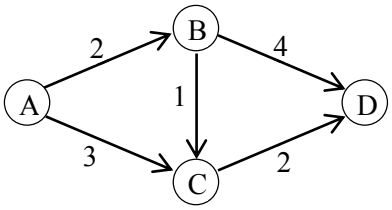
$$\begin{aligned} &\text{(P)} \ \min_{\boldsymbol{x}} \ z = \boldsymbol{c}^T\boldsymbol{x}, \ \text{ subject to } \ \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}, \boldsymbol{x} \ge \boldsymbol{0} \\ &\text{(D)} \ \max_{\boldsymbol{y}} \ w = \boldsymbol{y}^T\boldsymbol{b}, \ \text{ subject to } \ \boldsymbol{y}^T\boldsymbol{A} \le \boldsymbol{c}^T \end{aligned} \tag{5.11}$$

Download free eBooks at bookboon.com

where Lagrange multipliers $\boldsymbol{y}$ for the equality constraints in the dual formulation are unrestricted in sign. To obtain the above dual formulation, we use the following equivalence: $\boldsymbol{Ax} = \boldsymbol{b} \Leftrightarrow \boldsymbol{Ax} \geq \boldsymbol{b}, -\boldsymbol{Ax} \geq -\boldsymbol{b}$, or: $[\boldsymbol{A} \quad -\boldsymbol{A}]\begin{bmatrix}\boldsymbol{x}\\ \boldsymbol{x}\end{bmatrix} \geq \begin{bmatrix}\boldsymbol{b}\\ -\boldsymbol{b}\end{bmatrix}$. We can then use the symmetric form of duality where the dual variable vector is given as: $[\boldsymbol{u}^T, \boldsymbol{v}^T]$. We obtain the above result by designating dual variables as: $\boldsymbol{y} = \boldsymbol{u} - \boldsymbol{v};\ \boldsymbol{u}, \boldsymbol{v} \geq \boldsymbol{0}$, where $\boldsymbol{y}$ is unrestricted in sign.

The following example is used to explain LP duality.

Example 5.6: Duality in LP problems



To illustrate duality, we consider the problem of sending goods from node A to node D in a simplified network (Pedregal, p. 45). Assuming that the total quantity to be shipped equals 1, let x_{ij} denote the fractional quantity to be shipped via link ij with associated transportation cost c_{ij} (shown in the figure). Then, the primal objective is to minimize the transportation costs and the primal problem is formulated as:

$$\min_{\boldsymbol{x}} z = 2x_{AB} + 3x_{AC} + x_{BC} + 4x_{BD} + 2x_{CD}$$

Subject to: $x_{AB} = x_{BC} + x_{BD},\ x_{AC} + x_{BC} = x_{CD},\ x_{BD} + x_{CD} = 1$ (equivalently, $x_{AB} + x_{AC} = 1$); $x_{AB}, x_{BC}, x_{AC}, x_{BD}, x_{CD} \geq 0$

Alternatively, we may consider y_I to be the price of goods at node *I*, and $y_I - y_A$, as the profit to be made in transferring the goods from *A* to *I*. Then, the dual objective is to maximize the profit at node *D*. Then, if we arbitrarily assign: $y_A = 0$, the dual formulation is given as:

$$\max_{\boldsymbol{y}} y_D$$

Subject to: $y_B \leq 2,\ y_C \leq 3,\ y_C - y_B \leq 1,\ y_D - y_B \leq 4,\ y_D - y_C \leq 2$

Finally, we note that both problems can be formulated in terms of following coefficient matrices:

$$A = \begin{bmatrix}1 & 0 & -1 & -1 & 0\\ 0 & 1 & 0 & 0 & -1\\ 0 & 0 & 0 & 1 & 1\end{bmatrix},\ b = \begin{bmatrix}0\\0\\1\end{bmatrix},\ c^T = [2 \quad 3 \quad 1 \quad 4 \quad 2].$$

Download free eBooks at bookboon.com

5.5.1 Fundamental Duality Properties

Duality theory confers fundamental properties on the optimization problem that relate the primal and dual linear programs. Specifically, these properties specify bounds on the two objectives and are useful in developing computational procedures to solve the primal and dual problems. These properties are stated below for the symmetric form of duality where (P) solves the maximization problem.

Weak Duality. Let $\boldsymbol{x}$ denote a feasible solution to (P) and $\boldsymbol{y}$ a feasible solution to (D), then, $\boldsymbol{y}^T\boldsymbol{b} \geq \boldsymbol{y}^T\mathbf{A}\boldsymbol{x} \geq \boldsymbol{c}^T\boldsymbol{x}$, i.e., $w(\boldsymbol{y}) \geq z(\boldsymbol{x})$. Further, the difference between these two objective functions, $\boldsymbol{b}^T\boldsymbol{y} - \boldsymbol{c}^T\boldsymbol{x}$, is referred to as the duality gap.

Optimality. If $\boldsymbol{x}$ is a feasible solution to (P) and a $\boldsymbol{y}$ feasible solution to (D), and, further, $\boldsymbol{c}^T\boldsymbol{x} = \boldsymbol{b}^T\boldsymbol{y}$, then $\boldsymbol{x}$ is an optimal solution to (P), and $\boldsymbol{y}$ an optimal solution to (D).

Unboundedness. If the primal (dual) problem is unbounded, then the dual (primal) problem is infeasible (i.e., the feasible region is empty). We note that this is a necessary consequence of weak duality.

Strong Duality. If the primal (dual) problem has a finite optimal solution, then so does the dual (primal) problem; further, these two optimums are equal, i.e., $w_{opt} = \boldsymbol{y}^T\boldsymbol{b} = \boldsymbol{y}^T\mathbf{A}\boldsymbol{x} = \boldsymbol{c}^T\boldsymbol{x} = z_{opt}$.

Download free eBooks at bookboon.com

Further, $\boldsymbol{x}$ if is the optimal solution to (P), then $\boldsymbol{y}^T = \boldsymbol{c}_B^T \boldsymbol{B}^{-1}$ is the optimal solution to (D), which can be seen from: $w = \boldsymbol{y}^T \boldsymbol{b} = \boldsymbol{c}_B^T \boldsymbol{B}^{-1} \boldsymbol{b} = \boldsymbol{c}_B^T x_{\boldsymbol{B}} = \boldsymbol{c}^T \boldsymbol{x} = z$.

We also note that the optimality of (P) implies the feasibility of (D), and vice versa. In particular, $x_{\boldsymbol{B}} \geq \boldsymbol{0}$ (or, $\boldsymbol{x} \geq \boldsymbol{0}$) implies primal feasibility and dual optimality; whereas, $\hat{\boldsymbol{c}}_N^T \geq \boldsymbol{0}$ (or, $\hat{\boldsymbol{c}} = \boldsymbol{c} - \boldsymbol{A}^T \boldsymbol{y} \geq \boldsymbol{0}$) implies primal optimality and dual feasibility.

Complementary Slackness. At the optimal point, we have: $\boldsymbol{x}^T \boldsymbol{c} = \boldsymbol{x}^T \boldsymbol{A}^T \boldsymbol{y}$, implying: $\boldsymbol{x}^T (\boldsymbol{c} - \boldsymbol{A}^T \boldsymbol{y}) = \sum_{\boldsymbol{j}} x_j (c - A^T y)_j = 0$, which shows that it is not possible to have both $x_j > 0$ and $(A^T y)_j < c_j$ at the optimum.

Thus, if the *j*th primal variable is basic, i.e., $x_j = 0$, then the th dual constraint is binding, i.e., $(A^T y)_j = c_j$; and, if the *j*th primal variable is non-basic, i.e., $x_j = 0$, then the *j*th dual constraint is non-binding, i.e., $(A^T y)_j < c_j$.

5.5.2 The Dual Simplex Method

The dual simplex method involves application of the simplex method to the dual problem. Complementary to the regular simplex algorithm that initializes with a valid BFS and moves through primal feasible solutions, the dual simplex algorithm initializes with and moves through dual feasible (primal infeasible) solutions. The dual simplex algorithm thus iterates outsides of the feasible region. As such, the dual simplex method provides a convenient alternative to the two-phase simplex method in the event the optimization problem has no feasible solution (Sec. 5.3.2).

To develop the dual simplex algorithm, we consider the minimization problem formulated with dual variables (5.10). We note that primal optimality $(\hat{\boldsymbol{c}} \geq \boldsymbol{0})$ corresponds to dual feasibility $(\boldsymbol{y}^T \boldsymbol{A} \geq \boldsymbol{c}^T)$, and primal feasibility $(\boldsymbol{x} \geq \boldsymbol{0})$ corresponds to dual optimality. We therefore assume that the objective function coefficients are positive and the rhs is partly negative (some $b_i < 0$). The dual simplex algorithm then proceeds in a similar fashion to the primal algorithm except that:

1. The points generated during dual simplex iterations are primal infeasible as some basic variables have negative values.
2. The solutions are always optimal in the sense that the reduced cost coefficients for nonbasic variables are non-negative.
3. An optimal is reached when a feasible solution with non-negative values for the basic variables has been found.

A tableau implementation of the dual Simplex algorithm proceeds as follows: after subtracting the surplus variables from GE constraints in (5.12) we multiply those constraints by –1 We then enter the constraints and the cost function coefficients in a tableau noting that the initial basic solution is infeasible.

Download free eBooks at bookboon.com

At each iteration, the pivot element in the dual simplex method is determined as follows:

1. A pivot row A_q^T is selected as the row that has the basic variable with most negative value.
2. The ratio test to select the pivot column is conducted as: $\min_i \left\{ \frac{c_j}{-A_{q,j}} : c_j > 0, A_{q,j} < 0 \right\}$.

The dual simplex algorithm terminates when the rhs has become non-negative.

5.5.3 Recovery of the Primal Solution

The final tableaus resulting from the application of simplex methods to the primal and dual problems are intimately related. In particular, the elements in the last row of the final dual tableau replicate the elements in the last column of the final primal tableau, and vice versa. This fact allows the recovery of primal solution from the final dual tableau. Let the dual problem be solved using standard simplex method, then the value of the *i*th primal variable equals the reduced cost coefficient of the slack or surplus variable associated with the ith dual constraint in the final dual tableau. In addition, if the dual variable is nonbasic, then its reduced cost coefficient equals the value of the slack or surplus variable for the corresponding primal constraint.

To reveal the above relationship, we re-consider the dual problem in (5.12), which, after subtracting surplus variables, is represented in the following equivalent form:

$$\min_{\boldsymbol{y}} \; w = \boldsymbol{y}^T \boldsymbol{b}$$
Subject to: $\boldsymbol{y}^T \boldsymbol{A} - \boldsymbol{I}\boldsymbol{s} = \boldsymbol{c}^T, \; \boldsymbol{y} \geq \boldsymbol{0}$

An initial tableau for the dual problem, with $\boldsymbol{s}$ as the basic variables, is given as:

Basic	$\boldsymbol{y}$	$\boldsymbol{s}$	Rhs
$\boldsymbol{s}$	$-\boldsymbol{A}^T$	$\boldsymbol{I}$	$-\boldsymbol{c}$
$-\boldsymbol{w}$	$\boldsymbol{b}^T$	$\boldsymbol{0}$	0

Assuming that the same order of the variables is maintained, the final tableau at the termination of dual simplex algorithm may be given as:

Basic	$\boldsymbol{x}$	$\boldsymbol{s}$	Rhs
$\boldsymbol{y_B}$	$\widetilde{\boldsymbol{A}}$	$\boldsymbol{S}$	$\tilde{\boldsymbol{c}}$
$-\boldsymbol{w}$	$\widehat{\boldsymbol{b}}^T$	$\boldsymbol{x}^T$	$-z^*$

Download free eBooks at bookboon.com

where we note that the primal variables appear in the last row under the slack/surplus variable columns. Further, the coefficients in the final tableau are related to those in the initial tableau as follows:

$$\widetilde{\boldsymbol{A}} = -\boldsymbol{S}\boldsymbol{A}^T, \;\; \tilde{\boldsymbol{c}} = -\boldsymbol{S}\boldsymbol{c}, \qquad \widehat{\boldsymbol{b}}^T = \boldsymbol{b}^T - \boldsymbol{x}^T\boldsymbol{A}^T, \qquad z^* = \boldsymbol{c}^T\boldsymbol{x} \tag{5.13}$$

The following examples illustrate the efficacy of the dual Simplex algorithm.

Example 5.7: Dual Simplex algorithm

We consider the dual of Example 5.1 where the original LP problem was defined as:

$$\max_{x_1,x_2} z = 3x_1 + 2x_2$$
Subject to: $2x_1 + x_2 \leq 12, \;\; 2x_1 + 3x_2 \leq 16; \;\; x_1 \geq 0, x_2 \geq 0$

Using the symmetric form of duality, the dual optimization problem is defined as:

$$\min_{y_1,y_2} w = 12y_1 + 16y_2$$
Subject to: $2y_1 + 2y_2 \geq 3, \;\; y_1 + 3y_2 \geq 2; \;\; y_1 \geq 0, y_2 \geq 0$

Download free eBooks at bookboon.com

We subtract surplus variables from the GE constraints and multiply them with –1 before entering them in the initial tableau. We then follow with dual simplex iterations. The resulting series of tableaus is given below:

Basic	y_1	y_2	s_1	s_2	Rhs
s_1	-2	-2	1	0	-3
s_2	-1	-3	0	1	-2
$-w$	12	16	0	0	0

EBV: y_1, LBV: s_1, pivot: (1,1)

Basic	y_1	y_2	s_1	s_2	Rhs
y_1	1	1	-1/2	0	3/2
s_2	0	-2	-1/2	1	-1/2
$-w$	0	4	6	0	-18

LBV: s_2, EBV: y_2, pivot: (2,2)

Basic	y_1	y_2	s_1	s_2	Rhs
y_1	1	0	-3/4	1/2	5/4
y_2	0	1	1/4	-1/2	¼
$-w$	0	0	5	2	-19

At this point the dual LP problem is solved and the optimal solution is: $y_1 = 1.25, y_2 = 0.25, w_{opt} = 19$. We note that the first feasible solution obtained above is also the optimal solution. We further note that:

a) The optimal value of the objective function for (D) is the same as the optimal value for (P).
b) The optimal values for the basic variables for (P) appear as reduced costs associated with non-basic variables in (D).

As an added advantage, the dual simplex method obviates the need for the two-phase simplex method to obtain a solution when an initial BFS is not readily available. This is illustrated by re-solving Example 5.3 using the dual simplex algorithm.

Example 5.8: Dual Simplex algorithm

We consider the dual problem of Example 5.3. The original LP problem is stated as:

$$\max_{x_1,x_2} z = 3x_1 + 2x_2$$

Subject to: $3x_1 + 2x_2 \geq 12,\ 2x_1 + 3x_2 \leq 16,\ x_1 \geq 0, x_2 \geq 0$

Download free eBooks at bookboon.com

The GE constraint in the problem is first multiplied by –1; the problem is then converted to dual problem using the symmetric form of duality. The dual optimization problem is given as:

$$\min_{y_1,y_2} z_1 = -12y_1 + 16y_2$$
Subject to: $-3y_1 + 2y_2 \geq 3, -2y_1 + 3y_2 \geq 1;\ \ y_1 \geq 0, y_2 \geq 0$

The series of tableaus leading to the optimal solution via the dual simplex method is given below:

Basic	y_1	y_2	s_1	s_2	Rhs
s_1	3	-2	1	0	-3
s_2	2	-3	0	1	-2
$-w$	-12	16	0	0	0

LBV: s_1, EBV: y_2, pivot: (1,2)

Basic	y_1	y_2	s_1	s_2	Rhs
y_2	-3/2	1	-1/2	0	3/2
s_2	-5/2	0	-3/2	1	5/2
$-w$	12	0	8	0	-24

At this point the dual LP problem is solved with the optimal solution: $y_1^* = 0,\ y_2^* = 1.5,\ w^* = 24$. We note that this is the same solution obtained for Example 5.3. We further note that the reduced costs for nonbasic variables match with the optimal values of the primal basic variables.

The final dual Simplex example involves a problem with equality constraints.

Example 5.9: Equality Constraints

We re-consider Example 5.4 where the optimization problem was given as:

$$\min_{x_1,x_2} z = 2x_1 + x_2$$
Subject to: $x_1 + x_2 = 3,\ 0 \leq x_1, x_2 \leq 2$

In order to solve this problem via the dual Simplex method, we replace the equality constraint with twin inequality constraints: $\{x_1 + x_2 = 3\} \leftrightarrow \{x_1 + x_2 \leq 3, x_1 + x_2 \geq 3\}$. Next, we multiply GE constraint with , and add slack variables to all inequalities. Finally, we identify: s_1, s_2, s_3, s_4 as basic variables, and construct an initial tableau for the dual simplex method. This is followed by two iterations of the dual simplex algorithm leading to the optimum. The resulting tableaus for the problem are given below:

Download free eBooks at bookboon.com

Basic	x_1	x_2	s_1	s_2	s_3	s_4	Rhs
s_1	1	1	1	0	0	0	3
s_2	-1	-1	0	1	0	0	-3
s_3	1	0	0	0	1	0	2
s_4	0	1	0	0	0	1	2
$-z$	2	1	0	0	0	0	0

LBV: s_2, EBV: x_2, pivot: (2,2)

Basic	x_1	x_2	s_1	s_2	s_3	s_4	Rhs
s_1	0	0	1	1	0	0	0
x_2	1	1	0	-1	0	0	3
s_3	1	0	0	0	1	0	2
s_4	-1	0	0	1	0	1	-1
$-z$	1	0	0	1	0	0	-3

LBV: s_4, EBV: x_1, pivot: (2,2)

Basic	x_1	x_2	s_1	s_2	s_3	s_4	Rhs
s_1	0	0	1	1	0	0	0
x_2	0	1	0	0	0	1	2
s_3	0	0	0	1	1	1	1
x_1	1	0	0	-1	0	-1	1
$-z$	0	0	0	2	0	1	-4

The dual Simplex algorithm terminates with $z_{opt} = 4$.

5.6 Non-Simplex Methods for Solving LP Problems

The non-simplex methods to solve LP problems include the interior-point methods that iterate through the interior of the feasible region, and attempt to decrease the duality gap between the primal and dual feasible solutions. These methods can have good theoretical efficiency and practical performance that is comparable with the simplex methods. In the following, we discuss the primal-dual interior-point method that has been particularly successful in the case of LP problems (Griva, Nash & Sofer, p. 321).

To introduce the primal-dual method, we consider asymmetrical form of duality where the primal and dual problems are described as:

$$\begin{aligned} &\text{(P)} \min_{\boldsymbol{x}} \; z = \boldsymbol{c}^T\boldsymbol{x} \\ &\quad \text{subject to: } \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b},\ \boldsymbol{x} \geq \boldsymbol{0} \\ &\text{(D)} \max_{\boldsymbol{x}} \; w = \boldsymbol{b}^T\boldsymbol{y} \\ &\quad \text{subject to: } \boldsymbol{A}^T\boldsymbol{y} + \boldsymbol{s} = \boldsymbol{b},\ \boldsymbol{s} \geq \boldsymbol{0} \end{aligned} \tag{5.14}$$

Download free eBooks at bookboon.com

We note that for $\boldsymbol{x}$ and $\boldsymbol{y}$ to be the feasible solutions to the primal and dual problems (at the optimum), they must satisfy the following complementary slackness condition: $x_j s_j = 0,\ j = 1, \ldots, n.$ The primal-dual method begins with $x_j s_j = \mu,$: for some $\mu > 0,$ and iteratively reduces the values of μ, generating a series of vectors: $\boldsymbol{x}(\mu), \boldsymbol{y}(\mu), \boldsymbol{s}(\mu)$ along the way, in an effort to reduce the duality gap: $\boldsymbol{c}^T\boldsymbol{x} - \boldsymbol{b}^T\boldsymbol{y} = n\mu.$.

To develop the primal-dual algorithm, let the updates to the current estimates: $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{s},$ be given as: $\boldsymbol{x} + \Delta\boldsymbol{x}, \boldsymbol{y} + \Delta\boldsymbol{y}, \boldsymbol{s} + \Delta\boldsymbol{s};$ then, these updates are required to satisfy the following feasibility and complementarity conditions: $\boldsymbol{A}(\boldsymbol{x} + \Delta\boldsymbol{x}) = \boldsymbol{b},\ \boldsymbol{A}^T(\boldsymbol{y} + \Delta\boldsymbol{y})(\boldsymbol{s} + \Delta\boldsymbol{s}) = \boldsymbol{c},\ (\boldsymbol{x} + \Delta\boldsymbol{x})^T(\boldsymbol{s} + \Delta\boldsymbol{s}) = \boldsymbol{0}.$ Accordingly,

$$\begin{aligned} &\boldsymbol{A}\Delta\boldsymbol{x} = \boldsymbol{0} \\ &\boldsymbol{A}^T\Delta\boldsymbol{y} + \Delta\boldsymbol{s} = \boldsymbol{0} \\ &\big(x_j + \Delta x_j\big)\big(s_j + \Delta s_j\big) \cong x_j s_j + x_j\Delta s_j + s_j\Delta x_j = \mu \end{aligned} \tag{5.15}$$

where the latter condition has been linearized for ease of implementation. To proceed further, we define: $\boldsymbol{X} = diag(\boldsymbol{x}),\ \boldsymbol{S} = diag(\boldsymbol{s}),\ \boldsymbol{e} = [1, \ldots, 1]^T,$ to express the complementarity condition as: $\boldsymbol{XSe} = \mu\boldsymbol{e}.$ Next, let $\boldsymbol{D} = \boldsymbol{S}^{-1}\boldsymbol{X},\ \boldsymbol{v}(\mu) = (\mu\boldsymbol{I} - \boldsymbol{XS})\boldsymbol{e},$ then a solution to the linear system (5.16) is given as:

$$\begin{aligned} &\Delta\boldsymbol{x} = \boldsymbol{S}^{-1}\boldsymbol{v}(\mu) - \boldsymbol{D}\Delta\boldsymbol{s} \\ &\Delta\boldsymbol{y} = -(\boldsymbol{A}\boldsymbol{D}\boldsymbol{A}^T)^{-1}\boldsymbol{A}\boldsymbol{S}^{-1}\boldsymbol{v}(\mu) \\ &\Delta\boldsymbol{s} = -\boldsymbol{A}^T\Delta\boldsymbol{y} \end{aligned} \tag{5.16}$$

Download free eBooks at bookboon.com

In practice, to ensure primal and dual feasibility, the following update rule for the solution vectors has been suggested (Griva, Nash & Sofer, p. 324):

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \alpha_k \Delta \boldsymbol{x}_k, \qquad \boldsymbol{y}_{k+1} = \boldsymbol{y}_k + \alpha_k \Delta \boldsymbol{y}_k, \qquad \boldsymbol{s}_{k+1} = \boldsymbol{s}_k + \alpha_k \Delta \boldsymbol{s}_k$$

$$\alpha_k < \min(\alpha_P, \alpha_D), \qquad \alpha_P = \min_{\Delta x_j < 0} -\frac{x_j}{\Delta x_j}, \qquad \alpha_D = \min_{\Delta s_j < 0} -\frac{s_j}{\Delta s_j}$$

Finally, an initial estimate that satisfies (5.9) is needed to start the primal-dual method. To find that estimate, let the constraint equation for the primal problem (5.7) be written as: $\boldsymbol{I}\boldsymbol{x}_B + \boldsymbol{Q}\boldsymbol{x}_N = \boldsymbol{b}$; then, for some $\boldsymbol{x}_0, \boldsymbol{y}_0$, a set of feasible vectors satisfying (5.9) is obtained as: $\boldsymbol{x} = \begin{bmatrix} \boldsymbol{x}_0 \\ \boldsymbol{b} - \boldsymbol{Q}\boldsymbol{x}_0 \end{bmatrix}$, $\boldsymbol{y} = \boldsymbol{y}_0$, $\boldsymbol{s} = \begin{bmatrix} \boldsymbol{c} - \boldsymbol{A}^T\boldsymbol{y}_0 \\ -\boldsymbol{y}_0 \end{bmatrix}$.

Further, the bounding parameter μ is updated in successive iterations as: $\mu_{k+1} = \gamma\mu_k$, $0 < \gamma < 1$, where $\gamma = 0.1$ is considered a reasonable choice.

The primal-dual algorithm is given as follows:

Primal-Dual Algorithm:

Given $\boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c}$
Initialize: select: $\epsilon > 0$, $\mu > 0$, $0 < \gamma < 1$, N (maximum number of iterations).
Find initial $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{s} > \boldsymbol{0}$ to satisfy (5.9).
For $k = 1,2, \ldots$

1. Check termination: if $\boldsymbol{x}^T\boldsymbol{s} - n\mu < \epsilon$, or if $k > N$, quit.
2. Use (5.16) to compute the updates vectors.
3. Use (5.17) to compute α_k and perform the update.
4. Set $\mu \longleftarrow \gamma\mu$.

An example of the primal-dual algorithm is presented below.

Example 5.10: Primal-Dual Algorithm

We re-consider Example 5.1 where the original optimization problem was given as:

$$\max_{x_1, x_2} z = 3x_1 + 2x_2$$
Subject to: $2x_1 + x_2 \leq 12$, $2x_1 + 3x_2 \leq 16$; $x_1 \geq 0, x_2 \geq 0$

The coefficient matrices for the problem are: $\boldsymbol{A} = \begin{bmatrix} 2 & 1 & 1 & 0 \\ 2 & 3 & 0 & 1 \end{bmatrix}$, $\boldsymbol{b} = \begin{bmatrix} 12 \\ 16 \end{bmatrix}$, $\boldsymbol{c}^T = [-3 \quad -2]$.

Download free eBooks at bookboon.com

To initialize the primal-dual algorithm, we select the following parameters:

$\boldsymbol{x}_0 = [2,\ 2]^T, \boldsymbol{y}_0 = [-1,-1]^T\ \epsilon = 10^{-6},\ \mu = 10,\ \gamma = 0.1,\ N = 10.$

Then, the initial estimates for the variables are: $x^T = [2,2,6,6],\ y^T = [-1,-1],\ s^T = [1,2,1,1].$

The variable updates for the first eight iterations are given below, where the last column contains the residual:

$\boldsymbol{x_1}$	$\boldsymbol{x_2}$	$\boldsymbol{s_1}$	$\boldsymbol{s_2}$	$\boldsymbol{x^T s - n\mu}$
5.6923	0.6154	0.0001	2.7693	22.0000
5.5478	0.6773	0.2272	2.8726	-95.2300
4.8849	2.0767	0.1535	0.0000	-2.3342
4.9945	1.9904	0.0205	0.0398	-1.2668
5.0008	1.9983	0.0000	0.0034	-0.0507
5.0000	2.0000	0.0001	0.0001	-0.0017
5.0000	2.0000	0.0000	0.0000	-0.0002
5.0000	2.0000	0.0000	0.0000	-0.0000

The optimum solution is given as: $x_1^* = 5.0, x_2^* = 2.0,$ which agrees with the results of Example 5.1.

Download free eBooks at bookboon.com

5.7 Optimality Conditions for LP Problems

This section discusses the application of FONC to the LP problems. The first order optimality conditions in the case of general optimization problems are known as the KKT conditions. For convex optimization problems, the KKT conditions are both necessary and sufficient for optimality.

5.7.1 KKT Conditions for LP Problems

To derive the KKT conditions for the LP problems, we consider a maximization problem proposed in (5.10) above. Using slack variables, the problem is converted into standard form as:

$$\begin{aligned} &\min_{\boldsymbol{x}} \; z = -\boldsymbol{c}^T\boldsymbol{x} \\ &\text{subject to } \boldsymbol{A}\boldsymbol{x} - \boldsymbol{b} + \boldsymbol{s} = \boldsymbol{0},\ \boldsymbol{x} \geq \boldsymbol{0} \end{aligned} \tag{5.18}$$

We now use Lagrange multiplier vectors $\boldsymbol{u}, \boldsymbol{v}$ for the constraints to write a Lagrangian function as:

$$\mathcal{L}(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{v}) = -\boldsymbol{c}^T\boldsymbol{x} - \boldsymbol{u}^T\boldsymbol{x} + \boldsymbol{v}^T(\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b} + \boldsymbol{s})$$

Then, the first order KKT conditions for the optimality of the solution vector are:

Feasibility: $\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b} + \boldsymbol{s} = \boldsymbol{0}$
Optimality: $\nabla\mathcal{L}(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{v}) = \boldsymbol{A}^T\boldsymbol{v} - \boldsymbol{c} - \boldsymbol{u} = \boldsymbol{0}$
Complementarity: $\boldsymbol{u}^T\boldsymbol{x} + \boldsymbol{v}^T\boldsymbol{s} = \boldsymbol{0}$
Non-negativity: $\boldsymbol{x} \geq \boldsymbol{0},\ \boldsymbol{s} \geq \boldsymbol{0},\ \boldsymbol{u} \geq \boldsymbol{0},\ \boldsymbol{v} \geq \boldsymbol{0}$

The above equations need to be simultaneously solved for the unknowns: $\boldsymbol{x}, \boldsymbol{s}, \boldsymbol{u}, \boldsymbol{v}$ to find the optimum. By substituting $\boldsymbol{s}, \boldsymbol{u}$ from the first two equations into the third, the optimality conditions are reduced to:

$$\boldsymbol{v}^T(\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}) = \boldsymbol{0}, \qquad \boldsymbol{x}^T(\boldsymbol{c} - \boldsymbol{A}^T\boldsymbol{v}) = \boldsymbol{0}, \qquad \boldsymbol{x} \geq \boldsymbol{0}, \qquad \boldsymbol{v} \geq \boldsymbol{0}$$

Therefore, the following duality conditions are implied at the optimum point:

a) Lagrange multipliers for the active (binding) constraints are positive, and
b) Dual constraints associated with basic variables are binding.

Alternatively, we can solve the optimality conditions by partitioning the problem into basic and nonbasic variables as: $\boldsymbol{x}^T = [\boldsymbol{x}_B^T,\ \boldsymbol{x}_N^T]; \boldsymbol{c}^T = [\boldsymbol{c}_B^T,\ \boldsymbol{c}_N^T];\ \boldsymbol{A} = [\boldsymbol{B},\ \boldsymbol{N}];\ \boldsymbol{u}^T = [\boldsymbol{u}_B^T,\ \boldsymbol{u}_N^T]$. Then, in terms of partitioned variables, the optimality conditions are given as:

$$\begin{bmatrix}\boldsymbol{B}^T \\ \boldsymbol{N}^T\end{bmatrix}\boldsymbol{v} - \begin{bmatrix}\boldsymbol{u}_B \\ \boldsymbol{u}_N\end{bmatrix} - \begin{bmatrix}\boldsymbol{c}_B \\ \boldsymbol{c}_N\end{bmatrix} = \begin{bmatrix}\boldsymbol{0} \\ \boldsymbol{0}\end{bmatrix}, \qquad [\boldsymbol{x}_B \;\; \boldsymbol{x}_N]\begin{bmatrix}\boldsymbol{u}_B \\ \boldsymbol{u}_N\end{bmatrix} = \boldsymbol{0}$$

Download free eBooks at bookboon.com

Since $\boldsymbol{x}_B \neq 0, \boldsymbol{u}_B = 0$. Then, from the first equation, $\boldsymbol{v}^T = \boldsymbol{c}_B^T \boldsymbol{B}^{-1}$, and from the second equation, $\boldsymbol{u}_N^T = \boldsymbol{c}_B^T \boldsymbol{B}^{-1}\boldsymbol{N} - \boldsymbol{c}_N^T = \hat{\boldsymbol{c}}_N^T$. Thus, the reduced cost coefficients for nonbasic variables are the Lagrange multipliers, which are required to be non-negative at the optimum, i.e., $\boldsymbol{u}_N > 0$.

We can extend the optimality conditions to the dual problem formulated in (5.10). For the symmetric form of duality, the KKT conditions for the primal and dual problems are given as (Belegundu and Chandrapatla, p. 161):

	Primal	Dual
Feasibility:	$\boldsymbol{Ax} + \boldsymbol{s} = \boldsymbol{b}$	$\boldsymbol{A}^T\boldsymbol{v} - \boldsymbol{u} = \boldsymbol{c}$
Optimality:	$\boldsymbol{c} = \boldsymbol{A}^T\boldsymbol{v} - \boldsymbol{u}$	$\boldsymbol{b} = \boldsymbol{Ax} + \boldsymbol{s}$
Complementarity:	$\boldsymbol{u}^T\boldsymbol{x} + \boldsymbol{v}^T\boldsymbol{s} = \boldsymbol{0}$	
Non-negativity:	$\boldsymbol{x} \geq \boldsymbol{0},\ \boldsymbol{s} \geq \boldsymbol{0},\ \boldsymbol{u} \geq \boldsymbol{0},\ \boldsymbol{v} \geq \boldsymbol{0}$	

We note that the optimality condition for (P) is equivalent to the feasibility condition for (D) and vice versa, i.e., by interchanging the feasibility and optimality conditions, we may view the problem as primal or dual. It also shows that if (P) is unbounded, then (D) is infeasible, and vice versa.

5.7.2 A Geometric Viewpoint

Further insight into the solution is obtained from geometrical consideration of the problem. Towards that end, let $\boldsymbol{A}$ be expressed in terms of row vectors as: $\boldsymbol{A}^T = [\boldsymbol{a}_1^T, \boldsymbol{a}_2^T, \ldots, \boldsymbol{a}_m^T]$, where $\boldsymbol{a}_1^T$ represents a vector normal to the constraint: $\boldsymbol{a}_i^T\boldsymbol{x} + s_i = b_i$. Similarly, let $-\boldsymbol{e}_j$ denote a vector normal to the non-negativity constraint: $-x_j \leq 0$. Then, the optimality requires that there exist real numbers, $v_i \geq 0, i = 1, \ldots, m$ and $u_j \geq 0, j = 1, \ldots, n$, such that the following conditions hold:

$$\begin{aligned} &\boldsymbol{c} = \textstyle\sum_i v_i\, \boldsymbol{a}_i^T - \sum_j u_j\, \boldsymbol{e}_j \\ &\textstyle\sum_i v_i s_i + \sum_j u_j x_j = 0 \end{aligned} \tag{5.21}$$

Let the Lagrange multipliers be grouped as: $\mu_i \in \{u_i, v_j\}$, and let $N^i \in \{\boldsymbol{a}_i^T, -\boldsymbol{e}_j\}$ denote the set of active constraint normals, then the complementarity condition is expressed as: $\boldsymbol{c} = -\nabla z = \sum_{i \in \mathcal{J}} \mu_i\, N^i$, where $\mathcal{J}$ denotes the set of active constraints.

The above condition states that at the optimal point the negative of objective function gradient lies in the convex cone spanned by the active constraint normals. When this condition holds, the descent-feasible cone is empty, i.e., we cannot move in a direction that further decreases the objective function without leaving the feasible region. This result is consistent with Farkas Lemma, which for the LP problems is stated as follows:

Download free eBooks at bookboon.com

Farka's Lemma (Belegundu and Chandrupatla, p. 204): Given a set of vectors, $\boldsymbol{a}_i, i = 1, \ldots, m,$ and a vector $\boldsymbol{c}$, there is no vector $\boldsymbol{d}$ satisfying the conditions $\boldsymbol{c}^T\boldsymbol{d} < 0$ and $\boldsymbol{a}_i^T\boldsymbol{d} > 0, i = 1, \ldots, m$, if and only if $\boldsymbol{c}$ can be written as: $\boldsymbol{c} = \sum_{i=1}^{m} \mu_i \boldsymbol{a}_i, \mu_i \geq 0$.

An illustrative example for the optimality conditions appears below:

Example 5.11: Optimality Conditions for the LP problem

We reconsider example 5.1 that was formulated as:

$$\max_{x_1,x_2} z = 3x_1 + 2x_2$$
Subject to: $3x_1 + 2x_2 \geq 12,\ 2x_1 + 3x_2 \leq 16,\ x_1 \geq 0, x_2 \geq 0$

Application of the optimality conditions results in the following equations:

$$x_1(2v_1 + 2v_2 - 2) + x_2(v_1 + 3v_2 - 3) = 0$$
$$v_1(2x_1 + x_2 - 12) + v_2(2x_1 + 3x_2 - 16) = 0$$

Download free eBooks at bookboon.com

We split these into four equations and use Matlab symbolic toolbox to solve them, which gives the following candidate solutions:

$$\{x_1, x_2, v_1, v_2\} = (0,0,0,0), (6,0,1,0), (8,0,0,1), (5,2,0,1), (0,12,3,0), (0,5.33,0,1), \left(8 - \frac{3z}{2}, z, 0,1\right)$$

Then, it can be verified that the optimality conditions hold only in the case of: $\{x_1, x_2, v_1, v_2\} = (5,2,01)$. The optimum value of the objective function is: z* = 17.

5.8 The Quadratic Programming Problem

Theory developed for the LP problems easily extends to quadratic programming (QP) problems. The QP problem arises frequently in convex programming when the energy associated with a problem is to be minimized. An example of that is the finite element analysis (FEA) problem in structures.

The QP problem involves minimization of a quadratic cost function subject to linear constraints, and is described as:

$$\begin{aligned} &\min\ q(\boldsymbol{x}) = \tfrac{1}{2}\boldsymbol{x}^T\boldsymbol{Q}\boldsymbol{x} + \boldsymbol{c}^T\boldsymbol{x} \\ &\text{Subject to: } \boldsymbol{A}\boldsymbol{x} \geq \boldsymbol{b},\ \boldsymbol{x} \geq \boldsymbol{0} \end{aligned} \tag{5.22}$$

where $\boldsymbol{Q}$ is symmetric positive semidefinite. We first note that the feasible region for the QP problem is convex; further, for the given condition on $\boldsymbol{Q}$, $q(\boldsymbol{x})$ is convex. Therefore, QP problem is a convex optimization problem, and the KKT conditions are both necessary and sufficient for a global solution.

5.8.1 Optimality Conditions for QP Problems

To derive KKT conditions for the QP problem, we consider a Lagrangian function that includes Lagrange multipliers $\boldsymbol{u}$, $\boldsymbol{v}$ for the non-negativity and inequality constraints. The Lagrangian function and its gradient are given as:

$$\begin{aligned} &\mathcal{L}(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{v}) = \tfrac{1}{2}\boldsymbol{x}^T\boldsymbol{Q}\boldsymbol{x} + \boldsymbol{c}^T\boldsymbol{x} - \boldsymbol{u}^T\boldsymbol{x} - \boldsymbol{v}^T(\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b} - \boldsymbol{s}) \\ &\nabla\mathcal{L}(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{v}) = \boldsymbol{Q}\boldsymbol{x} + \boldsymbol{c} - \boldsymbol{u} - \boldsymbol{A}^T\boldsymbol{v} \end{aligned} \tag{5.23}$$

where $\boldsymbol{s}$ is a vector of slack variables. The resulting KKT conditions for the QP problem are given as:

Feasibility: $\boldsymbol{A}\boldsymbol{x} - \boldsymbol{s} = \boldsymbol{b}$
Optimality: $\boldsymbol{Q}\boldsymbol{x} + \boldsymbol{c} - \boldsymbol{u} - \boldsymbol{A}^T\boldsymbol{v} = \boldsymbol{0}$
Complementarity: $\boldsymbol{u}^T\boldsymbol{x} + \boldsymbol{v}^T\boldsymbol{s} = \boldsymbol{0}$
Non-negativity: $\boldsymbol{x} \geq \boldsymbol{0},\ \boldsymbol{s} \geq \boldsymbol{0},\ \boldsymbol{u} \geq \boldsymbol{0},\ \boldsymbol{v} \geq \boldsymbol{0}$

Download free eBooks at bookboon.com

By eliminating variables $\boldsymbol{s}$, $\boldsymbol{u}$ we can concisely express the KKT conditions as:

$$\boldsymbol{x}^T(\boldsymbol{Q}\boldsymbol{x}+\boldsymbol{c}-\boldsymbol{A}^T\boldsymbol{v})=\boldsymbol{0}, \qquad \boldsymbol{v}^T(\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b})=\boldsymbol{0}, \quad \boldsymbol{x}\geq\boldsymbol{0}, \qquad \boldsymbol{v}\geq\boldsymbol{0} \tag{5.24}$$

Alternatively, we combine the optimality and feasibility conditions in matrix form as:

$$\begin{bmatrix}\boldsymbol{Q} & -\boldsymbol{A}^T\\ \boldsymbol{A} & \boldsymbol{0}\end{bmatrix}\begin{bmatrix}\boldsymbol{x}\\ \boldsymbol{v}\end{bmatrix}+\begin{bmatrix}\boldsymbol{c}\\ -\boldsymbol{b}\end{bmatrix}-\begin{bmatrix}\boldsymbol{u}\\ \boldsymbol{s}\end{bmatrix}=\begin{bmatrix}\boldsymbol{0}\\ \boldsymbol{0}\end{bmatrix}$$

Next, let: $\boldsymbol{M}=\begin{bmatrix}\boldsymbol{Q} & -\boldsymbol{A}^T\\ \boldsymbol{A} & \boldsymbol{0}\end{bmatrix}$, $\boldsymbol{z}=\begin{bmatrix}\boldsymbol{x}\\ \boldsymbol{v}\end{bmatrix}$, $\boldsymbol{w}=\begin{bmatrix}\boldsymbol{u}\\ \boldsymbol{s}\end{bmatrix}$, $\boldsymbol{q}=\begin{bmatrix}\boldsymbol{c}\\ -\boldsymbol{b}\end{bmatrix}$; then, the problem is transformed as: $\boldsymbol{M}\boldsymbol{z}+\boldsymbol{q}=\boldsymbol{w}$, where the complementarity conditions are: $\boldsymbol{w}^T\boldsymbol{z}=\boldsymbol{0}$. The resulting problem is known in linear algebra as the Linear Complementarity Problem (LCP) and is solved in Sec. 5.8 below.

The above QP problem may additionally include linear equality constraints of the form: $\boldsymbol{C}\boldsymbol{x}=\boldsymbol{d}$, in which case, the problem is defined as:

$$\begin{aligned}&\min\ q(\boldsymbol{x})=\tfrac{1}{2}\boldsymbol{x}^T\boldsymbol{Q}\boldsymbol{x}+\boldsymbol{c}^T\boldsymbol{x}\\ &\text{subject to}\ \ \boldsymbol{A}\boldsymbol{x}\geq\boldsymbol{b},\ \boldsymbol{C}\boldsymbol{x}=\boldsymbol{d},\ \boldsymbol{x}\geq\boldsymbol{0}\end{aligned} \tag{5.26}$$

We similarly add slack variables to the inequality constraint, and formulate the Lagrangian function as:

$$\mathcal{L}(\boldsymbol{x},\boldsymbol{u},\boldsymbol{v})=\tfrac{1}{2}\boldsymbol{x}^T\boldsymbol{Q}\boldsymbol{x}+\boldsymbol{c}^T\boldsymbol{x}-\boldsymbol{v}^T(\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b}-\boldsymbol{s})-\boldsymbol{u}^T\boldsymbol{x}+\boldsymbol{w}^T(\boldsymbol{C}\boldsymbol{x}-\boldsymbol{d})$$

The modified KKT conditions are given as:

Feasibility: $\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b}-\boldsymbol{s}=\boldsymbol{0},\ \boldsymbol{C}\boldsymbol{x}=\boldsymbol{d}$
Optimality: $\boldsymbol{Q}\boldsymbol{x}+\boldsymbol{c}-\boldsymbol{u}-\boldsymbol{A}^T\boldsymbol{v}+\boldsymbol{C}^T\boldsymbol{w}=\boldsymbol{0}$
Complementarity: $\boldsymbol{u}^T\boldsymbol{x}+\boldsymbol{v}^T\boldsymbol{s}=\boldsymbol{0}$
Non-negativity: $\boldsymbol{x}\geq\boldsymbol{0},\ \boldsymbol{s}\geq\boldsymbol{0},\ \boldsymbol{u}\geq\boldsymbol{0},\ \boldsymbol{v}\geq\boldsymbol{0}$

where the Lagrange multipliers $\boldsymbol{w}$ for the equality constraints are not restricted in sign. By introducing: $\boldsymbol{w}=\mathbf{y}-\mathbf{z};\ \ \mathbf{y},\mathbf{z}\geq\boldsymbol{0}$, we can represent the combined optimality and feasibility conditions as:

$$\begin{bmatrix}\boldsymbol{Q} & -\boldsymbol{A}^T\\ \boldsymbol{A} & \boldsymbol{0}\\ \boldsymbol{C} & \boldsymbol{0}\end{bmatrix}\begin{bmatrix}\boldsymbol{x}\\ \boldsymbol{v}\end{bmatrix}-\begin{bmatrix}\boldsymbol{I} & \boldsymbol{0}\\ \boldsymbol{0} & \boldsymbol{I}\\ \boldsymbol{0} & \boldsymbol{0}\end{bmatrix}\begin{bmatrix}\boldsymbol{u}\\ \boldsymbol{s}\end{bmatrix}+\begin{bmatrix}\boldsymbol{C}^T & -\boldsymbol{C}^T\\ \boldsymbol{0} & \boldsymbol{0}\\ \boldsymbol{0} & \boldsymbol{0}\end{bmatrix}\begin{bmatrix}\boldsymbol{y}\\ \boldsymbol{z}\end{bmatrix}+\begin{bmatrix}\boldsymbol{c}\\ -\boldsymbol{b}\\ -\boldsymbol{d}\end{bmatrix}=\begin{bmatrix}\boldsymbol{0}\\ \boldsymbol{0}\\ \boldsymbol{0}\end{bmatrix} \tag{5.28}$$

The above problem can be similarly solved via LCP framework, which is introduced in Sec. 5.8.

Download free eBooks at bookboon.com

5.8.2 The Dual QP Problem

We reconsider the QP problem (5.22) and observe that the Lagrangian function (5.23) is stationary at the optimum with respect to $\boldsymbol{x}$, $\boldsymbol{u}$, $\boldsymbol{v}$. Then, as per Lagrangian duality (Sec. 4.5), it can be used to define the following dual QP problem (called Wolfe's dual):

$$\begin{aligned} &\max_{\boldsymbol{x},\boldsymbol{u},\boldsymbol{v}} \mathcal{L}(\boldsymbol{x},\boldsymbol{u},\boldsymbol{v}) = \tfrac{1}{2}\boldsymbol{x}^T\boldsymbol{Q}\boldsymbol{x} + \boldsymbol{c}^T\boldsymbol{x} - \boldsymbol{u}^T\boldsymbol{x} + \boldsymbol{v}^T(\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}) \\ &\text{Subject to: } \nabla\mathcal{L}(\boldsymbol{x},\boldsymbol{u},\boldsymbol{v}) = \boldsymbol{Q}\boldsymbol{x} + \boldsymbol{c} - \boldsymbol{u} + \boldsymbol{A}^T\boldsymbol{v} = \boldsymbol{0},\ \boldsymbol{x} \geq \boldsymbol{0},\ \boldsymbol{v} \geq \boldsymbol{0} \end{aligned} \tag{5.29}$$

Further, by relaxing the non-negativity condition on the design variable $\boldsymbol{x}$, we can eliminate $\boldsymbol{u}$ from the formulation, which results in a simpler dual problem defined as:

$$\begin{aligned} &\max_{\boldsymbol{x},\boldsymbol{v}\geq\boldsymbol{0}} \mathcal{L}(\boldsymbol{x},\boldsymbol{v}) = \tfrac{1}{2}\boldsymbol{x}^T\boldsymbol{Q}\boldsymbol{x} + \boldsymbol{c}^T\boldsymbol{x} + \boldsymbol{v}^T(\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}) \\ &\text{Subject to: } \boldsymbol{Q}\boldsymbol{x} + \boldsymbol{c} + \boldsymbol{A}^T\boldsymbol{v} = \boldsymbol{0} \end{aligned} \tag{5.30}$$

The implicit function theorem allows us to express the solution vector $\boldsymbol{x}$ in the vicinity of the optimum point as a function of the Lagrange multipliers $\boldsymbol{v}$ as: $\boldsymbol{x} = \boldsymbol{x}(\boldsymbol{v})$. Next, the Lagrangian is expressed as an implicit function $\Phi(\boldsymbol{v})$ of the multipliers, termed as the dual function. Further, the dual function is obtained as a solution to the following minimization problem:

$$\Phi(\boldsymbol{v}) = \min_{\boldsymbol{x}} \mathcal{L}(\boldsymbol{x},\boldsymbol{v}) = \tfrac{1}{2}\boldsymbol{x}^T\boldsymbol{Q}\boldsymbol{x} + \boldsymbol{c}^T\boldsymbol{x} + \boldsymbol{v}^T(\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}) \tag{5.31}$$

Download free eBooks at bookboon.com

The solution is obtained by solving the FONC, the constraint in (5.30), for $\boldsymbol{x}$ as:

$$\boldsymbol{x}(\boldsymbol{v}) = -\boldsymbol{Q}^{-1}(\boldsymbol{A}^T\boldsymbol{v} + \boldsymbol{c}) \tag{5.32}$$

and substituting it in the Lagrangian function to obtain:

$$\begin{aligned}\Phi(\boldsymbol{v}) &= -\tfrac{1}{2}(\boldsymbol{A}^T\boldsymbol{v} + \boldsymbol{c})^T\boldsymbol{Q}^{-1}(\boldsymbol{A}^T\boldsymbol{v} + \boldsymbol{c}) - \boldsymbol{v}^T\boldsymbol{b} \\ &= -\tfrac{1}{2}\boldsymbol{v}^T(\boldsymbol{A}\boldsymbol{Q}^{-1}\boldsymbol{A}^T)\boldsymbol{v} - (\boldsymbol{c}^T\boldsymbol{Q}^{-1}\boldsymbol{A}^T + \boldsymbol{b}^T)\boldsymbol{v} - \tfrac{1}{2}\boldsymbol{c}^T\boldsymbol{Q}^{-1}\boldsymbol{c}\end{aligned} \tag{5.33}$$

In terms of the dual function, the dual QP problem is defined as:

$$\max_{\boldsymbol{v}\geq\boldsymbol{0}} \Phi(\boldsymbol{v}) = -\tfrac{1}{2}(\boldsymbol{A}^T\boldsymbol{v} + \boldsymbol{c})^T\boldsymbol{Q}^{-1}(\boldsymbol{A}^T\boldsymbol{v} + \boldsymbol{c}) - \boldsymbol{v}^T\boldsymbol{b} \tag{5.34}$$

The dual problem can also be solved by application of FONC, where the gradient and Hessian of $\Phi(\boldsymbol{v})$ are given as:

$$\nabla\Phi = -\boldsymbol{A}\boldsymbol{Q}^{-1}(\boldsymbol{A}^T\boldsymbol{v} + \boldsymbol{c}) - \boldsymbol{b}, \qquad \nabla^2\Phi = -\boldsymbol{A}\boldsymbol{Q}^{-1}\boldsymbol{A}^T \tag{5.35}$$

By solving $\nabla_{\boldsymbol{v}}\Phi = 0$, we obtain the solution to the Lagrange multipliers as:

$$\boldsymbol{v} = -(\boldsymbol{A}\boldsymbol{Q}^{-1}\boldsymbol{A}^T)^{-1}(\boldsymbol{A}^T\boldsymbol{Q}^{-1}\boldsymbol{c} + \boldsymbol{b}) \tag{5.36}$$

where the non-negativity of $\boldsymbol{v}$ is implied. Finally, the solution to the design variables is obtained from (5.32) as:

$$\boldsymbol{x} = \boldsymbol{Q}^{-1}\boldsymbol{A}^T(\boldsymbol{A}\boldsymbol{Q}^{-1}\boldsymbol{A}^T)^{-1}(\boldsymbol{A}^T\boldsymbol{Q}^{-1}\boldsymbol{c} + \boldsymbol{b}) - \boldsymbol{Q}^{-1}\boldsymbol{c} \tag{5.37}$$

The dual methods have been successfully applied in structural mechanics. As an example of the dual QP problem, we consider a one-dimensional finite element analysis (FEA) problem involving two nodes.

Example 5.10: Finite Element Analysis (Belegundu and Chandrupatla, p. 187)

Let q_1, q_2 represent nodal displacements in the simplified two node structure, and assume that a load P, where $P = 60kN$, is applied at node 1. The FEA problem is formulated as minimization of the potential energy function given as:

$$\min_{\boldsymbol{q}} \prod = \frac{1}{2}\boldsymbol{q}^T\boldsymbol{K}\boldsymbol{q} - \boldsymbol{q}^T\boldsymbol{f}$$
Subject to: $q_2 \leq 1.2$

In the above problem, $\boldsymbol{q}^T = [q_1, q_2]$ represents the vector of nodal displacements.

Download free eBooks at bookboon.com

The stiffness matrix $\boldsymbol{K}$ for the problem is given as: $\boldsymbol{K} = \frac{10^5}{3}\begin{bmatrix} 2 & -1 \\ -1 & 1 \end{bmatrix}\frac{N}{m}$.

For this problem: $\boldsymbol{Q} = \boldsymbol{K},\ \boldsymbol{f} = [P,\ 0]^T,\ \boldsymbol{c} = -\boldsymbol{f},\ \boldsymbol{A} = [0 \quad 1],\ \boldsymbol{b} = 1.2$.

Further, $\boldsymbol{AQ}^{-1}\boldsymbol{A}^T = 6 \times 10^{-5},\ \boldsymbol{c}^T\boldsymbol{Q}^{-1}\boldsymbol{A}^T = -1.8,\ \boldsymbol{c}^T\boldsymbol{Q}^{-1}\boldsymbol{c} = 1.08 \times 10^{-5}$.

We use (5.33) to obtain the dual function as: $\Phi(\boldsymbol{v}) = -3 \times 10^{-5}v^2 - 0.6v - 1.08 \times 10^{-5}$.
From (5.36) the solution to Lagrange multiplier is: $v = 1 \times 10^4$.
Then, from (5.37), the optimum solution to the design variables is: $q_1 = 1.5\ mm,\ q_2 = 1.2\ mm$.
The optimum value of potential energy function is: $\prod = 129\ Nm$.

Next, we proceed to define and solve the Linear Complementarity Problem.

5.9 The Linear Complementary Problem

The application of optimality conditions to LP and QP problems leads to the Linear Complementary Problem (LCP), which can be solved using Simplex based methods. The LCP aims at finding vectors that satisfy linear equality, non-negativity, and complementarity conditions. When used in the context of optimization, the LCP simultaneously solves both primal and dual problems.

The general LCP problem is defined as follows: Given a real symmetric positive definite matrix $\boldsymbol{M}$ and a vector, $\boldsymbol{q}$, find a vector $\boldsymbol{z} \geq \boldsymbol{0}$, such that: $\boldsymbol{w} = \boldsymbol{Mz} + \boldsymbol{q} \geq \boldsymbol{0},\ \boldsymbol{w}^T\boldsymbol{z} = \boldsymbol{0}$.

In the case of QP problem, we define: $\boldsymbol{M} = \begin{bmatrix} \boldsymbol{Q} & -\boldsymbol{A}^T \\ \boldsymbol{A} & \boldsymbol{0} \end{bmatrix},\ \boldsymbol{z} = \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{v} \end{bmatrix},\ \boldsymbol{w} = \begin{bmatrix} \boldsymbol{u} \\ \boldsymbol{s} \end{bmatrix},\ \boldsymbol{q} = \begin{bmatrix} \boldsymbol{c} \\ -\boldsymbol{b} \end{bmatrix}$, to cast the problem into the LCP framework. Further, if $\boldsymbol{Q}$ is positive semidefinite, so is $\boldsymbol{M}$, resulting in a convex LCP, which can be solved by Simplex methods, in particular, the Lemke's algorithm.

Toward finding a solution to the LCP, we observe that if all $q_i \geq 0$, then $\boldsymbol{z} = \boldsymbol{0}$. $\boldsymbol{z} = \boldsymbol{0}$ solves the LCP. It is, therefore, assumed that one or more $q_i < 0$. Lemke's algorithm introduces an artificial variable, z_0, where $z_0 = |\min(q_i)|$, to cast LCP into Phase I Simplex framework. The resulting problem is given as:

$$\min\ z_0$$

$$\text{Subject to: } \boldsymbol{Iw} - \boldsymbol{Mz} - \boldsymbol{e}z_0 = \boldsymbol{q},\ \boldsymbol{w}^T\boldsymbol{z} = \boldsymbol{0},\ \boldsymbol{w} \geq \boldsymbol{0},\ \boldsymbol{z} \geq \boldsymbol{0} \tag{5.38}$$

Download free eBooks at bookboon.com

where $\boldsymbol{e} = [1\ 1\ \cdots 1]^T$, and $\boldsymbol{I}$ is an identity matrix. The linear constraint is used to define the starting tableau for the Simplex method, where an initial BFS is given as: $\boldsymbol{w} = \boldsymbol{q} + \boldsymbol{e}z_0 \geq 0,\ \boldsymbol{z} = 0$. The algorithm starts with a pivot operation aimed to bring z_0 into the basis. Thereafter, the EBV is selected as complement of the LBV in the previous iteration. Thus, if w_r leaves the basis, z_r enters the basis in the next tableau, or vice versa, which maintains the complementarity condition $w_r z_r = 0$. The algorithm terminates when z_0 has become nonbasic.

Lemke's Algorithm for solving LCP (Belegundu and Chandrupatla, p. 178):

1. If all $q_i > 0$, then LCP solution is: $z_0 = 0, \boldsymbol{w} = \boldsymbol{q}, \boldsymbol{z} = \boldsymbol{0}$. No further steps are necessary.
2. If some $q_i < 0$, select $z_0 = |\min(q_i)|$ to construct the initial tableau.
3. Choose the most negative q_i row and the z_0 column to define the pivot element. In the first step z_0 enters the basis, w_i corresponding to most negative q_i, exits the basis. Henceforth, all $q_i \geq 0$.
4. If basic variable in column *i* last exited the basis, its complement in column *j* enters the basis. (At first iteration, w_i exits and z_i enters the basis). Perform the ratio test for column *j* to find the least among q_i /(positive row element *i*). The basic variable corresponding to row *i* now exits the basis. If there are no positive row elements, there is no solution to the LCP
5. If the last operation results in the exit of the basic variable z_0, then the cycle is complete, stop. Otherwise go to step 3.

Download free eBooks at bookboon.com

Two examples of Lemke's algorithm are presented below:

Example 5.11: Lemke's algorithm

We consider the following QP problem:

$$\min_{x_1,x_2} f(x_1,x_2) = x_1^2 + x_2^2 - x_1x_2 - x_1 + 2x_2$$
Subject to: $x_1 + x_2 \le 5, x_1 + 2x_2 \le 10; x_1, x_2 \ge 0$

For the given problem: $\boldsymbol{Q} = \begin{bmatrix} 2 & -1 \\ -1 & 2 \end{bmatrix}$, $\boldsymbol{c}^T = [-1 \quad 2]$, $\boldsymbol{A} = \begin{bmatrix} 1 & 1 \\ 1 & 2 \end{bmatrix}$, $\boldsymbol{b} = \begin{bmatrix} 5 \\ 10 \end{bmatrix}$, $z_0 = -1$.

The resulting initial tableau for the problem is given as:

Basic	w_1	w_2	w_3	w_4	z_1	z_2	z_3	z_4	z_0	q
w_1	1	0	0	0	-2	1	-1	-1	-1	-1
w_2	0	1	0	0	1	-2	-1	-2	-1	2
w_3	0	0	1	0	1	1	0	1	-1	5
w_4	0	0	0	1	1	2	0	0	-1	10

pivot(1,9)

We begin by a pivot step aimed at bringing z_0 into the basis as represented by the following tableau:

Basic	w_1	w_2	w_3	w_4	z_1	z_2	z_3	z_4	z_0	q
z_0	-1	0	0	0	2	-1	1	1	1	1
w_2	-1	1	0	0	3	-3	0	-1	0	3
w_3	-1	0	1	0	3	0	1	1	0	2
w_4	-1	0	0	1	3	1	1	1	0	3

Pivot(1,5)

This is followed by further simplex iterations that maintain the complementarity conditions. The algorithm terminates when exits the basis. The resulting series of tableaus is given below:

Basic	w_1	w_2	w_3	w_4	z_1	z_2	z_3	z_4	z_0	q
z_1	-0.5	0	0	0	1	-0.5	0.5	0.5	0.5	0.5
w_2	0.5	1	0	0	0	-1.5	-1.5	-2.5	-1.5	1.5
w_3	0.5	0	1	0	0	1.5	-0.5	-0.5	-1.5	0.5
w_4	0.5	0	0	1	0	2.5	-0.5	-0.5	-1.5	1.5

The algorithm terminates after two steps as z_0 has exited the basis. The basic variables are given as: z_1, w_2, w_3, w_4, so that the complementarity conditions are satisfied, and the optimum solution is given as: $x_1 = 0.5, x_2 = 0, f^* = -0.25$.

Download free eBooks at bookboon.com

As the second LCP example, we reconsider the one-dimensional finite element analysis (FEA) problem that was solved earlier (Example 5.8).

Example 5.12: Finite Element Analysis (Belegundu and Chandrupatla, p. 187)

The problem is stated as:

$$\min_{\boldsymbol{q}} \prod = \frac{1}{2}\boldsymbol{q}^T\boldsymbol{K}\boldsymbol{q} - \boldsymbol{q}^T\boldsymbol{f}$$

Subject to: $q_2 \leq 1.2$

In the above problem, $\boldsymbol{q}^T = [q_1, q_2]$ represents a vector of nodal displacements. A load P, $P = 60kN$, is applied at node 1, so that $\boldsymbol{f} = [P, 0]^T$.The stiffness matrix $\boldsymbol{K}$ is given as: $\boldsymbol{K} = \frac{10^5}{3}\begin{bmatrix} 2 & -1 \\ -1 & 1 \end{bmatrix}\frac{N}{m}$.
For this problem: $\boldsymbol{Q} = \boldsymbol{K},\ \boldsymbol{c} = -\boldsymbol{f},\ \boldsymbol{A} = [0 \quad 1],\ \boldsymbol{b} = 1.2,\ z_0 = -1$.

The initial and the subsequent tableaus leading to the solution of the problem are given below:

Basic	$\boldsymbol{w_1}$	$\boldsymbol{w_2}$	$\boldsymbol{w_3}$	$\boldsymbol{z_1}$	$\boldsymbol{z_2}$	$\boldsymbol{z_3}$	$\boldsymbol{z_0}$	$\boldsymbol{q}$
$\boldsymbol{w_1}$	1	0	0	-66667	33333	0	-1	-60000
$\boldsymbol{w_2}$	0	1	0	33333	-33333	-1	-1	0
$\boldsymbol{w_3}$	0	0	1	0	1	0	-1	1.2

Pivot(1,7)

Basic	$\boldsymbol{w_1}$	$\boldsymbol{w_2}$	$\boldsymbol{w_3}$	$\boldsymbol{z_1}$	$\boldsymbol{z_2}$	$\boldsymbol{z_3}$	$\boldsymbol{z_0}$	$\boldsymbol{q}$
$\boldsymbol{z_0}$	-1	0	0	66667	-33333	0	1	60000
$\boldsymbol{w_2}$	-1	1	0	100000	-66667	-1	0	60000
$\boldsymbol{w_3}$	-1	0	1	66667	-33332	0	0	60001.2

Pivot(2,4)

Basic	$\boldsymbol{w_1}$	$\boldsymbol{w_2}$	$\boldsymbol{w_3}$	$\boldsymbol{z_1}$	$\boldsymbol{z_2}$	$\boldsymbol{z_3}$	$\boldsymbol{z_0}$	$\boldsymbol{q}$
$\boldsymbol{z_0}$	-0.33	-0.67	0	0	11111	0.67	1	20000
$\boldsymbol{z_1}$	-0.00001	0.00001	0	1	-0.67	-0.00001	0	0.6
$\boldsymbol{w_3}$	-0.33	-0.67	1	0	11112	0.67	0	20001.2

Pivot(3,5)

Download free eBooks at bookboon.com

Basic	w_1	w_2	w_3	z_1	z_2	z_3	z_0	q
z_0	-0.00003	-0.00006	-0.9999	0	0	0.00006	1	0.59995
z_1	-0.00003	-0.00003	0.0006	1	0	0.00003	0	1.79996
z_2	-0.00003	-0.00006	0.0009	0	1	0.00006	0	1.79995

Pivot(1,6)

Basic	w_1	w_2	w_3	z_1	z_2	z_3	z_0	q
z_3	-0.5	-1	-16667	0	0	1	16668	10000
z_1	-0.000015	0	0.5	1	0	0	-0.4	1.5
z_2	-0.0	0	1	0	1	0.0	-1	1.2

The algorithm terminates when z_0 has exited the basis. The final solution to the problem is given as: $z_1 = 1.5\ mm, z_2 = 1.2\ mm,\ \prod = 129\ Nm.$

Download free eBooks at bookboon.com

6 Discrete Optimization

This chapter is devoted to the study of solution approaches to discrete optimization problems that involve decision making, when the variables must be chosen from a discrete set. Many real world design problems fall in this category. For example, variables in optimization problems arising in production or transportation of goods represent discrete quantities and can only take on integer values. Further, scheduling and networking problems (e.g., assigning vehicles to transportation networks, frequency assignment in cellular phone networks, etc.) are often modeled with variables that can only take on binary values. The integer programming problem and binary integer programming problem are special cases of optimization problems where solution choices are limited to discrete sets.

Discrete optimization is closely related to combinatorial optimization that aims to search for the best object from a set of discrete objects. Classical combinatorial optimization problems include the econometric problems (knapsack problem, capital budgeting problem), scheduling problems (facility location problem, fleet assignment problem) and network and graph theoretic problems (traveling salesman problem, minimum spanning tree problem, vertex/edge coloring problem, etc.). Combinatorial optimization problems are NP-complete, meaning they are non-deterministic polynomial time problems, and finding a solution is not guaranteed in finite time. Heuristic search algorithms are, therefore, commonly employed to solve combinatorial optimization problems. Considerable research has also been devoted to finding computation methods that utilize polyhedral structure of integer programs.

Learning Objectives. The learning aims in this chapter are:

1. Study the structure and formulation of a discrete optimization problem.
2. Learn common solution approaches to the discrete optimization problems.
3. Learn to use the branch-and-bound and cutting plane methods to solve the mixed integer programming problem.

6.1 Discrete Optimization Problems

A discrete optimization problem may be formulated in one of the following ways:

1. An integer programming (IP) problem is formulated as:
   $\max_{\boldsymbol{x}} z = \boldsymbol{c}^T\boldsymbol{x}$,
   subject to $\boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b},\ \boldsymbol{x} \in \mathbb{Z}^n,\ \boldsymbol{x} \geq \boldsymbol{0}$

Download free eBooks at bookboon.com

2. A binary integer programming (BIP) problem is formulated as:
   $\max_{\boldsymbol{x}} z = \boldsymbol{c}^T\boldsymbol{x}$,
   subject to $\boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b},\ \boldsymbol{x} \in \{0,1\}^n$

3. A combinatorial optimization (CO) problem is formulated as:
   $\max_{\boldsymbol{x}} z = \boldsymbol{c}^T\boldsymbol{x}$,
   subject to $\boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b},\ x_i \in \{0,1\}\ (i \in B), x_i \in \mathbb{Z}\ (i \in I)$

4. A Mixed integer programming (MIP) problem is formulated as:
   $\max_{\boldsymbol{x}} z = \boldsymbol{c}^T\boldsymbol{x}$,
   subject to $\boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b},\ x_i \geq 0, x_i \in \mathbb{Z}, i = 1, \dots, n_d;\ x_{iL} \leq x_i \leq x_{iU}, i = n_d + 1, \dots, n$

5. A general mixed variable design optimization problem is formulated as:
   $\min_{\boldsymbol{x}} f(\boldsymbol{x})$,
   subject to $h_i(\boldsymbol{x}) = 0,\ i = 1, \dots, l;\ g_j(x) \leq 0, j = i, \dots, m;\ x_i \in D,\ i = 1, \dots, n_d;\ x_{iL} \leq x_i \leq x_{iU}, i = n_d + 1, \dots, n$

In the following, we discuss solution approaches to linear discrete optimization problems (1–4 above).

6.2 Solution Approaches to Discrete Problems

We first note that the discrete optimization problems may be solved by enumeration, i.e., an ordered listing of all solutions. The number of combinations to be evaluated to solve the problem is given as: $N_c = \prod_{i=1}^{n_d} q_i$, where n_d is the number of design variables and q_i represents the number of discrete values for the design variable x_i. This approach is, however, not practically feasible as the N_c increases rapidly with increase in n_d and q_i.

Further, two common approaches to solve linear discrete optimization problems are:

1. The branch and bound (BB) technique that divides the problem into a series of subprograms, where any solution to the original problem is contained in exactly one of the subprograms.
2. The cutting plane method that iteratively refines the solution by adding additional linear inequality constraints (cuts) aimed at excluding non-integer solutions to the problem.

These two approaches are discussed below. Besides, other approaches for solving discrete optimization problems include heuristic methods, such as tabu (neighborhood) search, hill climbing, simulated annealing, genetic algorithms, evolutionary programming, and particle swarm optimization. These topics are, however, not discussed here.

Download free eBooks at bookboon.com

In the following, we begin with the methods to solve an LP problem involving integral coefficients, followed by the BIP problems, and finally the IP/MIP problems.

6.3 Linear Programming Problems with Integral Coefficients

In this section, we consider an LP problem modeled with integral coefficients described as:

$$\begin{aligned} &\min_{\boldsymbol{x}} \; z = \boldsymbol{c}^T \boldsymbol{x} \\ &\text{Subject to } \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b},\ \boldsymbol{x} \geq \boldsymbol{0},\ \boldsymbol{A} \in \mathbb{Z}^{m\times n}, \boldsymbol{b} \in \mathbb{Z}^m, \boldsymbol{c} \in \mathbb{Z}^n \end{aligned} \tag{6.1}$$

We further assume that $\boldsymbol{A}$ is totally unimodular, i.e., every square submatrix $\boldsymbol{C}$ of $\boldsymbol{A}$, has $\det(\boldsymbol{C}) \in \{0, \pm 1\}$. In that case, every vertex of the feasible region, or equivalently, every BFS of the LP problem is integral. In particular, the optimal solution returned by the Simplex algorithm is integral. Thus, total unimodularity of A is a sufficient condition for integral solution to LP problems.

To show that an arbitrary BFS, $\boldsymbol{x}$, to the problem is integral, let $\boldsymbol{x}_B$ represent the elements of $\boldsymbol{x}$ corresponding to the basis columns, then there is a square nonsingular submatrix $\boldsymbol{B}$ of $\boldsymbol{A}$, such that $\boldsymbol{B}\boldsymbol{x}_B = \boldsymbol{b}$. Further, by unimodularity assumption, $\det(\boldsymbol{B}) = \pm 1$, and $\boldsymbol{B}^{-1} = \pm Adj\ \boldsymbol{B}$, where *Adj* represents the adjoint matrix and is integral. Therefore, $\boldsymbol{x}_B = \boldsymbol{B}^{-1}\boldsymbol{b}$ is integral.

Download free eBooks at bookboon.com

Further, if $\boldsymbol{A}$ is totally unimodular, so is $[\boldsymbol{A}\ \boldsymbol{I}]$. This applies to problems involving inequality constraints: $\boldsymbol{Ax} \leq \boldsymbol{b}$, which, when converted to equality via addition of slack variable $\boldsymbol{s}$ are represented as: $\boldsymbol{Ax} + \boldsymbol{Is} = \boldsymbol{b}$,or $[\boldsymbol{A}\ \boldsymbol{I}]\begin{bmatrix}\boldsymbol{x}\\ \boldsymbol{s}\end{bmatrix} = \boldsymbol{b}$. Then, if $\boldsymbol{A} \in \mathbb{Z}^{m\times n}$ is totally unimodular and $\boldsymbol{b} \in \mathbb{Z}^{m}$, all BFSs to the problem have integral components.

We, however, note that total unimodularity of $\boldsymbol{A}$ is a sufficient but not necessary condition for an integral solution; integral BFS may still be obtained if A is not totally unimodular. An example would be a matrix with isolated individual elements that do not belong to the set: $\{-1,0,1\}$.. Indeed, a necessary condition for integral solutions to LP problem is that each $m \times m$ basis submatrix $\boldsymbol{B}$ of $\boldsymbol{A}$ has determinant equal to ± 1.

An example of an LP problem with integral coefficients is considered below.

Example 6.1: Integer BFS

We consider the following LP problem with integer coefficients:

$$\max_{\boldsymbol{x}}\ z = 2x_1 + 3x_2$$
$$\text{Subject to: } x_1 \leq 3, x_2 \leq 5, x_1 + x_2 \leq 7, \boldsymbol{x} \in \mathbb{Z}^5, \boldsymbol{x} \geq \boldsymbol{0}$$

Following the introduction of slack variables, the constraint matrix and the right hand side are given as: $\boldsymbol{A} = \begin{bmatrix}1 & 0 & 1 & 0 & 0\\ 0 & 1 & 0 & 1 & 0\\ 1 & 1 & 0 & 0 & 1\end{bmatrix}, \boldsymbol{b} = \begin{bmatrix}3\\5\\7\end{bmatrix} \in \mathbb{Z}^3,$ where we note that $\boldsymbol{A}$ is unimodular and $\boldsymbol{b} \in \mathbb{Z}^3$. Then, using the simplex method, the optimal integral solution is obtained as: $\boldsymbol{x}^T = [2,5,1,0,0]$, with $z^* = 19$**.**

6.4 Binary Integer Programming Problems

In this section, we discuss solution of the BIP problem defined as:

$$\min_{\boldsymbol{x}}\ z = \boldsymbol{c}^T\boldsymbol{x}$$
$$\text{Subject to } \boldsymbol{Ax} \geq \boldsymbol{b},\ x_i \in \{0,1\},\ i = 1, \dots, n \tag{6.2}$$

where we additionally assume that $\boldsymbol{c} \geq \boldsymbol{0}$. We note that this is not a restrictive assumption, as any variable x_i with negative c_i in the objective function can be replaced by: $x_i' = 1 - x_i$.

Further, we note that under not-too-restrictive assumptions most LP problems can be reformulated in the BIP framework. For example, if the number of variables is small, and the bounds $x_{min} < x_i < x_{max}$ on the design variables are known, then each x_i can be represented as a binary number using k bits, where $2^{k+1} \geq x_{max} - x_{min}$. The resulting problem involves selection of the bits and is a BIP problem.

Download free eBooks at bookboon.com

The BIP problem can be solved by implicit enumeration. In implicit enumeration, obviously infeasible solutions are eliminated and the remaining ones are evaluated (i.e., enumerated) to find the optimum. The search starts from $x_i = 0,\ i = 1, \ldots, n,$ which is optimal. If this is not feasible, then we systematically adjust individual variable values till feasibility is attained. The implicit enumeration procedure is coded in the following algorithm that does not require an LP solver:

Binary Integer Programming Algorithm (Belegundu and Chandrupatla, p. 364):

1. Initialize: set $x_i = 0,\ i = 1, \ldots, n;$ if this solution is feasible, we are done.
2. For some $\boldsymbol{i}$ set $x_i = 1$. If the resulting solution is feasible, then record it if this is the first feasible solution, or if it improves upon a previously recorded feasible solution.
3. Backtrack (set $x_i = 0$) if a feasible solution was reached in the previous step, or if feasibility appears impossible in this branch.
4. Choose another *i* and return to 2.

The progress of the algorithm is graphically recorded in a decision-tree, using nodes and arcs with node 0 representing the initial solution $(x_i = 0,\ i = 1, \ldots, n)$, and node *i* representing a change in the value of variable x_i. From node *k*, if we choose to raise variable x_i to one, then this is represented as an arc from node *k* to node *i*. At node *i* the following possibilities exist:

1. The resulting solution is feasible, meaning no further improvement in this branch is possible.
2. Feasibility is impossible from this branch.
3. The resulting solution is not feasible, but feasibility or further improvement are possible.

In the first two cases, the branch is said to have been fathomed. We then backtrack to node *k*, where variable x_i is returned to zero. We next seek another variable to be raised to one. The algorithm continues till all branches have been fathomed, and returns an optimum 0-1 solution.

We consider the following example of a BIP problem.

Example 6.2: Implicit enumeration (Belegundu and Chandrupatla, p. 367)

A machine shaft is to be cut at two locations to given dimensions 1, 2, using one of the two available processes, A and B. The following information on process cost and three-sigma standard deviation is available, where the combined maximum allowable tolerance is to be limited to 12mils:

Download free eBooks at bookboon.com

Process\Job	Job1		Job2	
	Cost	SD	Cost	SD
Process A	$65	±4mils	$57	±5mils
Process B	$42	±6mils	$20	±10mils

Let $x_i, i = 1 - 4$ denote the available processes for both jobs, and let t_i denote their associated tolerances. The BIP problem is formulated as:

$\min_{x} z = 65x_1 + 57x_2 + 42x_3 + 20x_4$

Subject to: $x_1 + x_2 = 1, x_3 + x_4 = 1, \sum_i t_i \leq 12, x_i \in \{0,1\}, i = 1, \ldots, 4.$

Download free eBooks at bookboon.com

The problem is solved via implicit enumeration; the resulting decision-tree is represented below:

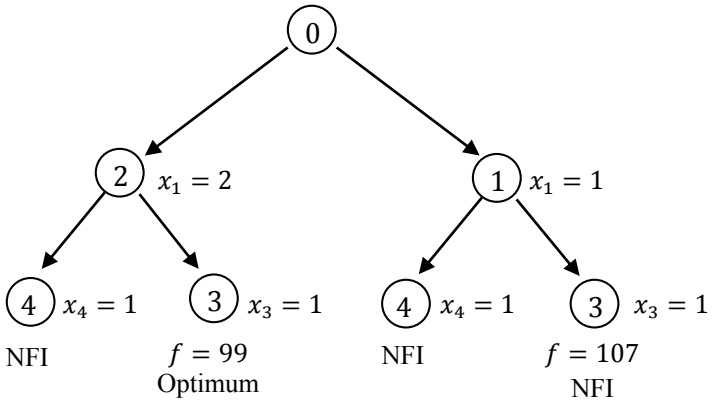


Fig 6.1: The decision tree for Example 6.2 (NFI: No further improvement

6.5 Integer Programming Problems

This section discusses the solution approaches to the IP problem formulated as:

$$\begin{aligned} &\max_{\boldsymbol{x}} \; z = \boldsymbol{c}^T \boldsymbol{x} \\ &\text{Subject to } \boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b},\ \boldsymbol{x} \in \mathbb{Z}^n,\ \boldsymbol{x} \geq \boldsymbol{0} \end{aligned} \tag{6.3}$$

To start with, the optimization problem that results when integrality constraint in the above problem is ignored is termed as LP relaxation of the IP problem. While a naïve solution to the IP problem may be to round off the non-integer LP relaxation solution, in general, this approach does not guarantee a satisfactory solution to IP problem.

In the following, we discuss two popular methods for solving IP problems: the branch and bound method and the cutting plane method. Both methods begin by first solving the LP relaxation problem and subsequently using the LP solution to subsequently bind the IP solution.

6.5.1 The Branch and Bound Method

The BB method is the most widely used method for solving IP problems. The method has its origins in computer science, where search over a large finite space is performed by using bounds on the objective function to prune the search tree.

Download free eBooks at bookboon.com

The BB method iteratively solves an IP problem as follows: it first obtains the LP relaxation solution; next, it introduces integrality constraints to define subprograms that effectively divide the feasible region into smaller subsets (branching); it then calculates objective function bounds for each subprogram (bounding); finally, it uses those bounds to discard non-promising solutions from further consideration (fathoming). The procedure ends when every branch has been fathomed and an optimum integer solution, if one exists, has been found.

A decision tree is normally used to record the progress of the BB algorithm, where the LP relaxation solution is represented as node 0. Subsequently, at each node *k, the algorithm sequences through the following phases:*

1. **Selection.** If some variables in the simplex solution at node *k* have non-integer values, the algorithm selects the one with the lowest index (or the one with greatest economic impact) for branching.
2. **Branching.** The solution at node k is partitioned into two mutually exclusive subsets, each represented by a node in the decision tree and connected to node k by an arc. It involves imposition of two integer constraints ($x_i \leq I,\ x_i \geq I+1,\ I = \lfloor x_i \rfloor$), generating two new subprograms where each solution to the original IP problem is contained in exactly one of the subprograms.
3. **Bounding.** In this phase, upper bounds on the optimal subproblem solutions are established. Solving a subprogram via LP solver results in one of the following possibilities:
    a) There is no feasible solution.
    b) The solution does not improve upon an available IP solution.
    c) An improved IP solution is returned and is recorded as current optimal.
    d) A non-integer solution that is better than the current optimal is returned.

4. **Fathoming.** In the first three cases above the current branch is excluded from further consideration. The algorithm then backtracks to the most recently unbranched node in the tree and continues with examining the next node in a last in first out (LIFO) search strategy.

Finally, the process ends when all branches have been fathomed, and an integer optimal solution to the problem, if one exists, has been found.

Let *NF* denote the set of nodes not yet fathomed, *F* denote the feasible region for the original IP problem, F_R denote the feasible region for the LP relaxation, F_k denote the feasible region at node k, S_k denote the subproblem defined as: $\max_{\boldsymbol{x}} z_k = \boldsymbol{c}^T\boldsymbol{x},\ \boldsymbol{x} \in F_k$, and let z_L denote the lower bound on the optimal solution. Then, the BB algorithm is given as follows:

Download free eBooks at bookboon.com

Branch-and-bound Algorithm (Sierksma, p. 219):

Initialize: set $F_0 = F_R,\ NF = \{0\},\ z_L = -\infty$..
While $NF \neq \emptyset$,

1. Select a label $k \in NF$..
2. Determine if there exists an optimal solution $(z_k, \boldsymbol{x}_k)$ to S_k, else set $z_k = -\infty$.
3. If $z_k > z_L$, then if $\boldsymbol{x}_k \in F$, set $z_L = z_k$.
4. If $z_k \leq z_L$, set $NF = NF \backslash \{k\}$.
5. If $z_k > z_L$ and $\boldsymbol{x}_k \notin F$, partition F_k into two or more subsets as follows: choose a variable $x_i \in \boldsymbol{x}_k$with fractional value, $x_i = I + \delta_i,\ I = \lfloor x_i \rfloor,\ 0 < \delta_i < 1$. Define two new subprograms: $F_{k_1} = F_k \cap \{x_i \leq I\},\ F_{k_2} = F_{k_2} \cap \{x_i \geq I + 1\}$. Set $NF = NF \cup \{k_1, k_2\}$.

An example is now presented to illustrate the BB algorithm.

Download free eBooks at bookboon.com

Example 6.3: Branch and bound algorithm

We consider the following IP problem (Belegundu and Chandrupatla, p. 383): A tourist bus company having a budget of $10M is considering acquiring a fleet with a mix of three models: a 15-seat van costing $35,000, a 30-seat minibus costing $60,000, and a 60-seat bus costing $140,000. A total capacity of 2000 seats is required. At least one third of the vehicles must be the big buses. If the estimated profits per seat per month for the three models are: $4, $3, and $2 respectively, determine the number of vehicles of each type to be acquired to maximize profit.

Let x_1, x_2, x_3 denote the quantities to be purchased for each of the van, minibus, and big bus; then, the optimization problem is formulated as:

Maximize $z = 60x_1 + 90x_2 + 120x_3$
Subject to: $35x_1 + 60x_2 + 140x_3 \leq 1000, 15x_1 + 30x_2 + 60x_3 \geq 2000, x_1 + x_2 - 2x_3 \leq 0;$
$x_1, x_2, x_3 \geq 0$ and integer

Following steps are taken to solve the problem. The progress is also shown in a decision tree in Fig. 6.2:

1. S_0: the LP relaxation problem $(F_0 = F_R)$ is first solved and produces an optimum solution: $x_1^* = 0, x_2^* = 7.69, x_3^* = 3.85,\ f^* = 1153.8$, which serves as an upper bound for IP solution.
2. $S_1: F_0 \cup \{x_3 \leq 3\}$ is solved and produces an integer solution: $x_1^* = 0, x_2^* = 6, x_3^* = 3,\ f^* = 900$. This is recorded as current optimum.
3. $S_2: F_0 \cup \{x_3 \geq 4\}$ produces a non-integer solution: $x_1^* = 1.6, x_2^* = 6.4, x_3^* = 4,\ f^* = 1152.$.
4. $S_3: F_2 \cup \{x_2 \leq 6\}$ produces a non-integer solution: $x_1^* = 2.1, x_2^* = 6, x_3^* = 4.05,\ f^* = 1151.4$.
5. $S_4: F_3 \cup \{x_3 \leq 4\}$ produces an integer solution: $x_1^* = 2, x_2^* = 6, x_3^* = 4,\ f^* = 1140$. This is recorded as the new optimum and the branch is fathomed.
6. $S_5: F_3 \cup \{x_3 \geq 5\}$ produces a non-integer solution: $x_1^* = 8.57, x_2^* = 0, x_3^* = 5,\ f^* = 1114.3$, which is lower than the current optimum, so the branch is fathomed.
7. $S_6: F_2 \cup \{x_2 \geq 7\}$ produces a non-integer solution: $x_1^* = 0.57, x_2^* = 7, x_3^* = 4,\ f^* = 1144.3$
8. $S_7: F_6 \cup \{x_1 \leq 0\}$ produces a non-integer solution: $x_1^* = 0, x_2^* = 7.33, x_3^* = 4,\ f^* = 1140$, which does not improve upon the current optimum. The branch is fathomed.
9. $S_8: F_6 \cup \{x_1 \geq 1\}$ has no feasible solution. The branch is fathomed.
10. All branches having been fathomed, the optimal solution is: $x^* = (2,6,4), f^* = 1140$.

Download free eBooks at bookboon.com

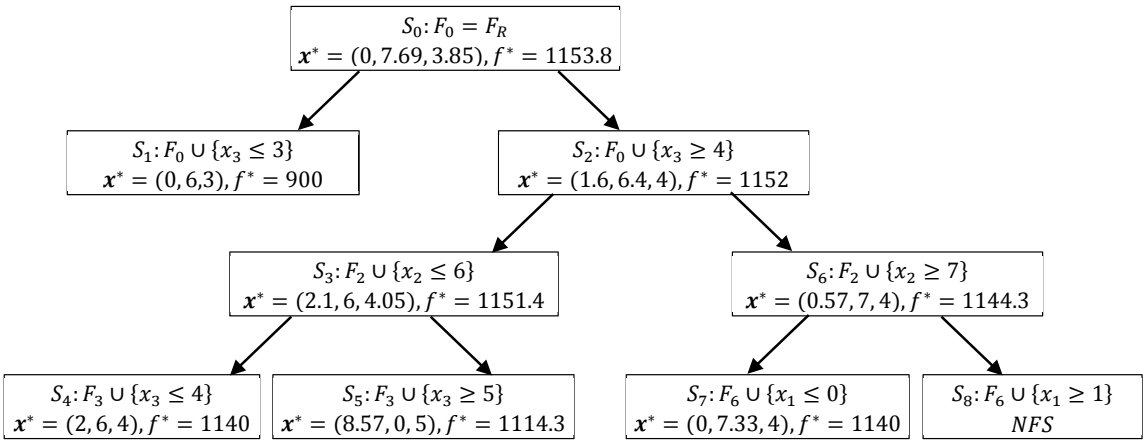


Fig. 6.2: The decision tree for Example 6.3.

6.5.2 The Cutting Plane Method

Proposed by Gomory in 1958, the cutting plane method or Gomory's method similarly begins with LP relaxation of the IP problem. It then trims the feasible region by successively adding linear constraints aimed to prune the non-integer solutions without losing any of the integer solutions. The new constraints are referred to as Gomory cuts. The process is repeated till an optimal integer solution has been obtained (Belegundu and Chandrupatla, p. 372; Chong and Zak, p. 438).

To develop the cutting plan method, we assume that the partitioned constraint matrix for the LP relaxation problem is given in canonical form as:

$$\boldsymbol{I}\boldsymbol{x}_B + \boldsymbol{A}_N\boldsymbol{x}_N = \boldsymbol{b} \tag{6.4}$$

where $\boldsymbol{x}_B$ and $\boldsymbol{x}_N$ refer to the basic and nonbasic variables. The corresponding BFS is given as: $\boldsymbol{x}_B = \boldsymbol{b}$, $\boldsymbol{x}_N = \boldsymbol{0}$. Next, we consider the ith component of the solution: $x_i + \sum_{j=m+1}^{n} a_{ij}x_j = b_i$, and use the floor operator to separate it into integer and non-integer parts as:

$$x_i + \sum_{j=m+1}^{n} \left(\lfloor a_{ij} \rfloor + \alpha_{ij}\right)x_j = \lfloor b_i \rfloor + \beta_i \tag{6.5}$$

Then, since $\lfloor a_{ij} \rfloor \leq a_{ij}$, a feasible solution that satisfies (6.5) also satisfies:

$$x_i + \sum_{j=m+1}^{n} \lfloor a_{ij} \rfloor x_j \leq b_i \tag{6.6}$$

Download free eBooks at bookboon.com

whereas, an integer feasible solution can be characterized by:

$$x_i + \sum_{j=m+1}^{n} \lfloor a_{ij} \rfloor x_j \leq \lfloor b_i \rfloor \tag{6.7}$$

The integer feasible solution also satisfies the difference of the two inequalities, which is given as:

$$\sum_{j=m+1}^{n} \alpha_{ij} x_j \geq \beta_i \tag{6.8}$$

The above inequality is referred to as the Gomory cut. We note that, since the left-hand-side equals zero, the optimal non-integer BFS does not satisfy this inequality. Thus, introduction of the inequality constraint (6.8) makes the current LP solution infeasible without losing any IP solutions.

The constraint introduced by Gomory cut is first brought into standard form by subtracting a surplus variable. The resulting problem is solved using simplex method for a new optimal BFS, which is then inspected for non-integer components. The process is repeated till an integer BFS has been obtained.

Download free eBooks at bookboon.com

The cutting plane algorithm generates a family of polyhedra which satisfy: $\Omega \supset \Omega_1 \supset \Omega_2 \supset \cdots \supset \Omega \cap \mathbb{Z}^n$, where $\Omega = \{x \in \mathbb{R}^n : \boldsymbol{Ax} \leq \boldsymbol{b}\}$ denote the polyhedral associated with the LP relaxation problem. The cutting plane algorithm terminates in finite steps.

An example of the cutting plane method is presented below.

Example 6.4: Cutting Plane method

We consider the IP problem in Example 6.3 above where the LP relaxation solution was found as: $x_1^* = 0, x_2^* = 7.69, x_3^* = 3.85,\ f^* = 1153.8$. The final tableau for the LP relaxation solution is given as:

Basic	$\boldsymbol{x_1}$	$\boldsymbol{x_2}$	$\boldsymbol{x_3}$	$\boldsymbol{s_1}$	$\boldsymbol{s_2}$	$\boldsymbol{s_3}$	Rhs
$\boldsymbol{x_2}$	0.808	1	0	0.539	0.039	0	7.69
$\boldsymbol{x_3}$	-0.096	0	1	-0.231	0.019	0	3.85
$\mathbf{s_3}$	0.173	0	0	0.115	0.115	1	123.1
$-\mathbf{z}$	0.115	0	0	2.077	0.577	0	1153.8

The following series of cuts then produces an integer optimum solution:

No.	Cut	Optimal solution
1.	$0.808x_1 + 0.539s_1 + 0.039s_2 - s_4 = 0.692$	$x_1^* = 0.857, x_2^* = 7, x_3^* = 3.929, f^* = 1152.9$
2.	$0.833s_1 + 0.024s_2 + 0.881s_4 - s_5 = 0.929$	$x_1^* = 2.162, x_2^* = 5.946, x_3^* = 4.054, f^* = 1151.3$
3.	$0.054s_1 + 0.973s_2 + 0.135s_5 - s_6 = 0.946$	$x_1^* = 2.083, x_2^* = 5.972, x_3^* = 4.028, f^* = 1145.8$
4.	$0.056s_1 + 0.139s_5 + 0.972s_6 - s_7 = 0.972$	$x_1^* = 2, x_2^* = 6, x_3^* = 4, f^* = 1140$

Download free eBooks at bookboon.com

7 Numerical Optimization Methods

This chapter describes the numerical methods used for solving both unconstrained and constrained optimization problems. These methods have been used to develop computational algorithms that form the basis of commercially available optimization software. The process of computationally solving the optimization problem is termed as mathematical programming and includes both linear and nonlinear programming. The basic numerical method to solve the nonlinear problem is the iterative solution method that starts from an initial guess, and iteratively refines it in an effort to reach the minimum (or maximum) of a multi-variable objective function. The iterative scheme is essentially a two-step process that seeks to determine: a) a search direction that does not violate the constraints and along which the objective function value decreases; and b) a step size that minimizes the function value along the chosen search direction. Normally, the algorithm terminates when either a minimum has been found, indicated by the function derivative being approximately zero, or when a certain maximum number of iterations has been exceeded indicating that there is no feasible solution to the problem.

Download free eBooks at bookboon.com

Learning Objectives: The learning objectives in this chapter are:

1. Understand numerical methods employed for solving optimization problems
2. Learn the approaches to numerically solve the line search problem in one-dimension
3. Learn the direction finding algorithms, including gradient and Hessian methods
4. Learn the sequential linear programming (SLP) and sequential quadratic programming (SQP) techniques

7.1 The Iterative Method

The general numerical optimization method begins with an initial guess and iteratively refines it so as to asymptotically approach the optimum. To illustrate the iterative method of finding a solution, we consider an unconstrained nonlinear programming problem defined as:

$$\min_{\boldsymbol{x}} \ f(\boldsymbol{x}) \tag{7.1}$$

where $\boldsymbol{x}$ denotes the set of optimization variables. Let $\boldsymbol{x}^k$ denote the current estimate of the minimum; then, the solution algorithm seeks an update, $\boldsymbol{x}^{k+1}$, that further reduces the function value, i.e., it results in: $f\left(\boldsymbol{x}^{k+1}\right) < f\left(\boldsymbol{x}^k\right)$, also termed as the descent condition.

In the general iterative scheme, the optimization variables are updated as per the following rule:

$$\boldsymbol{x}^{k+1} = \boldsymbol{x}^k + \alpha_k \boldsymbol{d}^k \tag{7.2}$$

In the above, $\boldsymbol{d}^k$ represents any search direction and α_k is the step size along that direction. The iterative method is thus a two-step process:

1. Find the suitable search direction $\boldsymbol{d}^k$ along which the function value locally decreases
2. Perform line search along $\boldsymbol{d}^k$ to find $\boldsymbol{x}^{k+1}$ such that $f\left(\boldsymbol{x}^{k+1}\right)$ attains its minimum value

We first consider the problem of finding a descent direction $\boldsymbol{d}^k$ and note that it can be determined by checking the directional derivative of $f\left(\boldsymbol{x}^k\right)$ along $\boldsymbol{d}^k$, which is given as the scalar product: $\nabla f\left(\boldsymbol{x}^k\right)^T \boldsymbol{d}^k$. Since the scalar product is a function of the angle between the two vectors, the descent condition is satisfied if the angle between $\nabla f\left(\boldsymbol{x}^k\right)$ and $\boldsymbol{d}^k$ is larger than 90°.

If the directional derivative of the function $f\left(\boldsymbol{x}^k\right)$ along $\boldsymbol{d}^k$ is negative, then the descent condition is satisfied. Further, $\boldsymbol{d}^k$ is a descent direction only if it satisfies: $\nabla f\left(\boldsymbol{x}^k\right)^T \boldsymbol{d}^k < 0$. If $\boldsymbol{d}^k$ If is a descent direction, then we are assured that at least for small positive values of α_k, $f\left(\boldsymbol{x}^k + \alpha_k \boldsymbol{d}^k\right) < f(\boldsymbol{x}^k)$.

Download free eBooks at bookboon.com

Next, assuming a suitable search direction $\boldsymbol{d}^k$ has been determined, we next seek to determine a suitable step length α_k, where an optimal value of α_k minimizes $f(\boldsymbol{x}^k)$ along $\boldsymbol{d}^k$. Since both $\boldsymbol{x}^k$ and $\boldsymbol{d}^k$ are known, the projected function value along $\boldsymbol{d}^k$ depends on α_k alone, and can be expressed as:

$$f(\boldsymbol{x}^k + \alpha_k \boldsymbol{d}^k) = f(\boldsymbol{x}^k + \alpha \boldsymbol{d}^k) = f(\alpha) \tag{7.3}$$

The problem of choosing α to minimize $f(\boldsymbol{x}^{k+1})$ along $\boldsymbol{d}^k$ thus amounts to a single-variable functional minimization problem, also known as the line search problem, and is defined as:

$$\min_{\alpha} f(\alpha) = f(\boldsymbol{x}^k + \alpha \boldsymbol{d}^k) \tag{7.4}$$

Assuming that a solution exists, it is found at a point where the derivative of the function goes to zero. Thus, by setting $f'(\alpha) = 0$, we can solve for the desired step size and update the current estimate $\boldsymbol{x}^k$.

As an example of the line search problem, we consider minimizing a quadratic function:

$$f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^T \boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}^T \boldsymbol{x}, \nabla f = \boldsymbol{A}\boldsymbol{x} - \boldsymbol{b} \tag{7.5}$$

where $\boldsymbol{A}$ is a symmetric positive definite matrix. Let $\boldsymbol{d}$ be a given descent direction; then, the line search problem reduces to the following minimization problem:

$$\min_{\alpha} f(\alpha) = (\boldsymbol{x}^k + \alpha \boldsymbol{d})^T \boldsymbol{A}(\boldsymbol{x}^k + \alpha \boldsymbol{d}) - \boldsymbol{b}^T(\boldsymbol{x}^k + \alpha \boldsymbol{d}) \tag{7.6}$$

A solution is found by setting $f'(\alpha) = \boldsymbol{d}^T \boldsymbol{A}(\boldsymbol{x}^k + \alpha \boldsymbol{d}) - \boldsymbol{d}^T \boldsymbol{b} = 0$, and is given as:

$$\alpha = \frac{\nabla f(\boldsymbol{x}^k)^T \boldsymbol{d}}{\boldsymbol{d}^T \boldsymbol{A}\boldsymbol{d}} = \frac{\boldsymbol{d}^T(\boldsymbol{A}\boldsymbol{x}^k - \boldsymbol{b})}{\boldsymbol{d}^T \boldsymbol{A}\boldsymbol{d}} \tag{7.7}$$

An update then follows as: $\boldsymbol{x}^{k+1} = \boldsymbol{x}^k + \alpha \boldsymbol{d}$.

In the following, we first discuss numerical methods used to solve the line search problem in Sec. 7.2, followed by a discussion of the methods to solve the direction finding problem in Sec. 7.3.

7.2 Computer Methods for Solving the Line Search Problem

In order to solve the line search problem, we assume that a suitable search direction $\boldsymbol{d}^k$ has been determined, and wish to minimize the objective function: $f(\boldsymbol{x}^k + \alpha \boldsymbol{d}^k) = f(\alpha)$ along $\boldsymbol{d}^k$. We further assume $\boldsymbol{d}^k$ that is a descent direction, i.e., it satisfies: $\nabla f(\boldsymbol{x}_k)^T \boldsymbol{d}^k < 0$, so that only positive values of α need to be considered. Then, the line search problem reduces to finding a solution to (7.4) above.

Download free eBooks at bookboon.com

In the following, we address the problem of finding the minimum of a function, $f(x),\ x \in \mathbb{R}$, where we additionally assume that the function is unimodal, i.e., it has a single local minimum. Prominent computer methods for solving the line search problem are described below.

7.2.1 Interval Reduction Methods

The interval reduction methods are commonly used to solve the line search problem. These methods find the minimum of a unimodal function in two steps:

a) Bracketing the minimum to an interval
b) Reducing the interval of uncertainty to desired accuracy

The bracketing step aims to find a three-point pattern, such that for $x_1, x_2, x_3,\ f(x_1) \leq f(x_2) > f(x_3)$. The bracketing algorithm can be started from any point in the domain of $f(x)$, though a good guess will reduce the number of steps involved. In the following description of the bracketing algorithm f_i denotes $f(x)$.

Download free eBooks at bookboon.com

Bracketing Algorithm (Belegundu & Chandrupatla p. 54):

1. Initialize: choose $x_1,\ \Delta,\ \gamma$ (e.g., $\gamma = 1.618$)
2. Set $x_2 = x_1 + \Delta$; evaluate f_1, f_2
3. If $f_1 < f_2$, set $x_0 \leftarrow x_1,\ x_1 \leftarrow x_2,\ x_2 \leftarrow x_0,\ \Delta = -\Delta$
4. Set $\Delta = \gamma\Delta$, $x_3 = x_2 + \Delta$; evaluate f_3
5. If $f_2 \geq f_3$, set $f_1 \leftarrow f_2,\ f_2 \leftarrow f_3,\ x_1 \leftarrow x_2,\ x_2 \leftarrow x_3$;, then go to step 3
6. Quit; points 1,2, and 3 satisfy $f_1 \geq f_2 < f_3$.

Next, we assume that the minimum has been bracketed to a closed interval $[x^l, x^u]$. The interval reduction step aims to find the minimum in that interval. A common interval reduction approach is to use either the Fibonacci or the Golden Section methods; both methods are based on the golden ratio derived from Fibonacci's sequence.

Fibonacci's Method. The Fibonacci's method uses Fibonacci numbers to achieve maximum interval reduction in a given number of steps. The Fibonacci number sequence is generated as: $F_0 = F_1 = 1,\ F_i = F_{i-1} + F_{i-2},\ i \geq 2$. Fibonacci numbers have some interesting properties, among them:

1. The ratio $\tau = \lim_{n\to\infty} \frac{F_{n-1}}{F_n} = \frac{\sqrt{5}-1}{2} = 0.618034$ is known as the golden ratio.
2. Using Fibonacci numbers, the number of interval reductions required to achieve a desired accuracy ε is the smallest n such that $1/F_n < \varepsilon$, and can be specified in advance.
3. For given l_1 and n, we have $I_2 = \frac{F_{n-1}}{F_n} I_1,\ I_3 = I_1 - I_2,\ I_4 = I_2 - I_3,$ etc.

The Fibonacci algorithm is given as follows:

Fibonacci Algorithm (Belegundu & Chandrupatla p. 60):

Initialize: specify $x_1, x_4\ (I_1 = |x_4 - x_1|),\ \varepsilon,\ n: \frac{1}{F_n} < \varepsilon$

Compute $\alpha_1 = \frac{F_{n-1}}{F_n}$; $x_2 = \alpha_1 x_1 + (1 - \alpha_1)x_4$, evaluate f_2

For $i = 1, \ldots, n-1$

1. Introduce $x_3 = (1 - \alpha_i)x_1 + \alpha_i x_4,$ evaluate f_3
2. If $f_2 < f_3$, set $x_4 \leftarrow x_1,\ x_1 \leftarrow x_3$
3. Else set $x_1 \leftarrow x_2,\ x_2 \leftarrow x_3, f_2 \leftarrow f_3$
4. Set $\alpha_{i+1} = \frac{I_{n-i-1}}{I_{n-i}}$

Download free eBooks at bookboon.com

Golden Section Method. The golden section method uses the golden ratio $\frac{I_{i+1}}{I_i} = \tau = 0.618034$ for interval reduction in the above Fibonacci algorithm. This results in uniform interval reduction strategy independent of the number of trials. Further, since the final interval I_n is related to the initial interval I_1 as: $I_n = \tau^{n-1} I_1$, given I_1 and a desired I_n, the number of interval reductions may be computed as: $n = \left\lfloor \frac{\ln I_n - \ln I_1}{\ln \tau} + \frac{3}{2} \right\rfloor$, where $\lfloor \cdot \rfloor$ represents the floor function.

The golden section method can be integrated with the three-point bracketing algorithm by choosing $\gamma = \frac{1}{\tau}$ and renaming x_3 as x_4. Stopping criteria for the golden section algorithm may be specified in terms of desired interval size, reduction in function value, or the number of interval reductions.

Next, the bracketing step can also be combined with the interval reduction step, and the integrated bracketing and interval reduction algorithm is given below.

Integrated Bracketing and Golden Section Algorithm (Belegundu & Chandrupatla p. 65):

Initialize: specify $x_1,\ \Delta,\ \tau = 0.618034,\ \varepsilon$

1. Set $x_2 = x_1 + \Delta$; evaluate f_1, f_2
2. If $f_1 < f_2$, set $x_0 \leftarrow x_1,\ x_1 \leftarrow x_2,\ x_2 \leftarrow x_0,\ \Delta = -\Delta$
3. Set $\Delta = \frac{\Delta}{\tau}$, $x_4 = x_2 + \Delta$; evaluate f_4
4. If $f_2 \geq f_4$, set $f_1 \leftarrow f_2,\ f_2 \leftarrow f_4,\ x_1 \leftarrow x_2,\ x_2 \leftarrow x_4$; then go to step 3
5. Introduce $x_3 = (1-\tau)x_1 + \tau x_4$, evaluate f_3
6. If $f_2 < f_3$, set $x_4 \leftarrow x_1,\ x_1 \leftarrow x_3$
7. Else set $x_1 \leftarrow x_2,\ x_2 \leftarrow x_3, f_2 \leftarrow f_3$
8. Check stopping criteria: If $|x_1 - x_3| < \varepsilon$, quit; else go to 5

7.2.2 Approximate Search Algorithms

The calculations of the exact step size in the line search step are time consuming. In most cases, approximate function minimization suffices to advance to the next iteration. Since crude minimization methods may give rise to convergence issues, additional conditions on both $\boldsymbol{d}^k$ and α_k are prescribed to ensure convergence of the numerical algorithm. These conditions include, for $\boldsymbol{d}^k$: a) sufficient descent condition, and b) gradient related condition; and for α_k: a) sufficient decrease condition, and b) non trivial condition. They are described below.

Download free eBooks at bookboon.com

Sufficient Descent Condition. The sufficient descent condition, or the angle condition guards against $\boldsymbol{d}^k$ becoming too close to $\nabla f(\boldsymbol{x}^k)$. The condition is normally stated as: $-\frac{\nabla f(\boldsymbol{x}^k)^T \boldsymbol{d}^k}{\|\nabla f(\boldsymbol{x}^k)\|\|\boldsymbol{d}^k\|} \geq \epsilon > 0$ for a small ϵ. Alternatively, the sufficient descent condition may be specified as: $\nabla f(\boldsymbol{x}^k)^T \boldsymbol{d}^k < c\|\nabla f(\boldsymbol{x}^k)\|^2, \ c > 0$.

Gradient Related Condition. The search direction is gradient related if $\|\boldsymbol{d}^k\| \geq c\|\nabla f(\boldsymbol{x}^k)\|, \ c > 0$. This condition aids in convergence.

Sufficient Decrease Condition. The sufficient decrease condition on α_k ensures that a nontrivial reduction in the function value is obtained at each step. The condition is derived from Taylor series expansion of $f(\boldsymbol{x}^k + \alpha_k \boldsymbol{d}^k)$ and is stated as: $f(\boldsymbol{x}^k + \alpha_k \boldsymbol{d}^k) - f(\boldsymbol{x}^k) \leq \mu \, \alpha_k \nabla f(\boldsymbol{x}^k)^T \boldsymbol{d}^k, \ 0 < \mu < 1$.

Arjimo's Rule. An alternative sufficient decrease condition, referred to as Arjimo's rule, is given as:

$$f(\alpha) \leq f(0) + \mu \alpha f'(0), \qquad 0 < \mu < 1 \tag{7.8}$$

Curvature Condition. A curvature condition is added to Arjimo's rule to improve convergence. The curvature condition is given as:

$$|f'(\alpha)| \leq \eta |f'(0)|, \qquad 0 \leq \eta < 1 \tag{7.9}$$

Download free eBooks at bookboon.com

Further, the curvature condition implies that: $\left|\nabla f\left(\boldsymbol{x}^k + \alpha_k \boldsymbol{d}^k\right)^T \boldsymbol{d}^k\right| \leq \eta \left|\nabla f\left(\boldsymbol{x}^k\right)^T \boldsymbol{d}^k\right|,\ 0 \leq \eta < 1.$

Conditions (7.8) and (7.9) together with $\mu \leq \eta$ are known as Wolfe conditions, which are commonly used by all line search algorithms. A line search based on Wolfe conditions proceeds by bracketing the minimizer in an interval, followed by estimating it via polynomial approximation. These two steps are explained below:

Bracketing the Minimum. In the bracketing step we seek an interval $\left[\underline{\alpha}, \overline{\alpha}\right]$ such that $f'(\underline{\alpha}) < 0$ and $f'(\overline{\alpha}) > 0$. Since for any descent direction, $f'(0) < 0$, therefore, $\alpha = 0$ serves as initial lower bound on α. To find an upper bound, increasing α values, e.g., $\alpha = 1,2, \ldots$, are tried. Assume that for some $\alpha_i > 0,\ f'(\alpha_i) < 0$ and $f'(\alpha_{i+1}) > 0$; then, α_i serves as an upper bound.

Estimating the Minimum. Once the minimum has been bracketed to a small interval, a quadratic or cubic polynomial approximation is used to find the minimizer. If the polynomial minimizer $\hat{\alpha}$ satisfies Wolfe's condition for the desired η value (say $\eta = 0.5$) and the sufficient decrease condition for the desired μ value (say $\mu = 0.2$), it is taken as the function minimizer, otherwise $\hat{\alpha}$ is used to replace one of the $\underline{\alpha}$ or $\overline{\alpha}$, and the polynomial approximation step repeated.

Quadratic curve Fitting. Assuming that the interval $[\alpha_l, \alpha_u]$ contains the minimum of a unimodal function, $f(\alpha)$, it can be approximated by a quadratic function: $q(\alpha) = a_0 + a_1\alpha + a_2\alpha^2$. A quadratic approximation uses three points $\{\alpha_l, \alpha_m, \alpha_u\}$, where the mid-point of the interval may be used for α_m. The quadratic coefficients $\{a_0, a_1, a_2\}$ are solved from: $f(\alpha_i) = a_0 + a_1\alpha_i + a_2\alpha_i^2,\ \ \alpha_i \epsilon \{\alpha_l, \alpha_m, \alpha_u\}$, which results in the following expressions:

$$\begin{aligned} a_2 &= \frac{1}{\alpha_u - \alpha_m}\left[\frac{f(\alpha_u) - f(\alpha_l)}{\alpha_u - \alpha_l} - \frac{f(\alpha_m) - f(\alpha_l)}{\alpha_m - \alpha_l}\right]; \\ a_1 &= \frac{1}{\alpha_m - \alpha_l}\left(f(\alpha_m) - f(\alpha_l)\right) - a_2(\alpha_l + \alpha_m); \\ a_0 &= f(\alpha_l) - a_1\alpha_l - a_2\alpha_l^2 \end{aligned} \tag{7.10}$$

The minimum for $q(\alpha)$ can be computed by setting $q'(\alpha) = 0,$ and is given as: $\alpha_{min} = -\frac{a_1}{2a_2}$. An explicit formula for α_{min} in terms of the three interval points can also be derived and is given as:

$$\alpha_{min} = \alpha_m - \frac{1}{2}\frac{(\alpha_m - \alpha_l)^2(f(\alpha_m) - f(\alpha_u)) - (\alpha_m - \alpha_u)^2(f(\alpha_m) - f(\alpha_l))}{(\alpha_m - \alpha_l)(f(\alpha_m) - f(\alpha_u)) - (\alpha_m - \alpha_u)(f(\alpha_m) - f(\alpha_l))} \tag{7.11}$$

An example of the approximate search algorithm is now presented.

Download free eBooks at bookboon.com

Example 7.1: Approximate search algorithm (Ganguli, p. 121)

We wish to approximately solve the following minimization problem: $\min_{\alpha} f(\alpha) = e^{-\alpha} + \alpha^2$. We use Arjimo's rule with: $\mu = 0.2$, and $\alpha = 0.1, 0.2, ...,$ to estimate the minimum. The Matlab commands used for this purpose and the corresponding results appear below:

```
>> f=inline('x.*x+exp(-x)'); mu=0.2; al=0:.1:1;
>> feval(f,al)
1.0000     0.9148     0.8587     0.8308     0.8303     0.8565
0.9088     0.9866     1.0893     1.2166     1.3679
>> 1-mu*al
1.0000     0.9800     0.9600     0.9400     0.9200     0.9000
0.8800
0.8600     0.8400     0.8200     0.8000
```

Then, according to Arjimo's condition, an estimate of the minimum is given as: $\alpha = 0.5$. Further, since $f'(0) < 0$ and $f'(\alpha) > 0$, and the minimum is bracketed by [0, 0.5]. We next use quadratic approximation of the function over $\{0, \frac{\alpha}{2}, \alpha\}$ to estimate the minimum as follows:

```
al=0; ai=0.25; au=0.5;
a2 = ((f(au)-f(al))/(au-al)-(f(ai)-f(al))/(ai-al))/(au-ai);
a1 = (f(ai)-f(al))/(ai-al)-a2*(al+ai);
xmin = -a1/a2/2 = 0.3531
```

An estimate of the minimum is given as: $\hat{\alpha} = 0.3531$. We note that the exact solution is given as: $\alpha_{min} = 0.3517$.

Next, we describe the computer methods for finding the search direction. Our initial focus is on unconstrained problems. The constrained problems are discussed later in Sec. 7.4.

7.3 Computer Methods for Finding the Search Direction

The computer methods for finding the search direction $\boldsymbol{d}^k$ are normally grouped into first order and second order methods, where the order refers to the derivative order of the function approximation used. Thus, first order methods refer to the gradient-based methods, while the second order methods additionally involve the Hessian matrix in the computations. The gradient based quasi-Newton methods are overwhelmingly popular when it comes to implementation. We describe popular search methods below.

Download free eBooks at bookboon.com

7.3.1 The Steepest Descent Method

The steepest descent method, attributed to Cauchy, is the simplest of the gradient methods. The method involves choosing $\boldsymbol{d}^k$ to locally move in the direction of maximum decrease in the function value, i.e., the direction opposite to the gradient vector at the current estimate point.

Thus, the steepest descent method is characterized by: $\boldsymbol{d}^k = -\nabla f(\boldsymbol{x}^k)$, which leads to the following update rule:

$$\boldsymbol{x}^{k+1} = \boldsymbol{x}^k - \alpha_k \cdot \nabla f(\boldsymbol{x}^k) \tag{7.12}$$

where the step size α_k to minimize $f(\boldsymbol{x}^{k+1})$ can be analytically or numerically determined using methods described in Sec. 7.2.

As an example, in the case of a quadratic function: $f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}^T\boldsymbol{x}, \nabla f = \boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}$, the steepest descent method with exact line search results in the following update rule:

$$\boldsymbol{x}^{k+1} = \boldsymbol{x}^k - \alpha \cdot \nabla f\left(\boldsymbol{x}^k\right); \;\; \alpha = \frac{\nabla f\left(\boldsymbol{x}^k\right)^T \nabla f\left(\boldsymbol{x}^k\right)}{\nabla f(\boldsymbol{x}^k)^T \mathbf{A} \nabla f(\boldsymbol{x}^k)} \tag{7.13}$$

Download free eBooks at bookboon.com

The above update can be equivalently described in terms of a residual: $\boldsymbol{r}_k = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_k = -\nabla f(\boldsymbol{x}^k)$ as:

$$\boldsymbol{x}^{k+1} = \boldsymbol{x}^k + \alpha_k \boldsymbol{r}_k; \;\; \alpha_k = \frac{\boldsymbol{r}_k^T \boldsymbol{r}_k}{\boldsymbol{r}_k^T A \boldsymbol{r}_k} \tag{7.14}$$

The steepest descent algorithm is given below.

Steepest Descent Algorithm:

Initialize: choose $\boldsymbol{x}^0$
For $k = 0,1,2, \ldots$

1. Compute $\nabla f(\boldsymbol{x}^k)$
2. Check convergence: if $\left\|\nabla f(\boldsymbol{x}^k)\right\| < \epsilon$, stop.
3. Set $\boldsymbol{d}^k = -\nabla f(\boldsymbol{x}^k)$
4. Line search problem: Find $\min_{\alpha \geq 0} f(\boldsymbol{x}^k + \alpha \boldsymbol{d}^k)$
5. Set $\boldsymbol{x}^{k+1} = \boldsymbol{x}^k + \alpha \boldsymbol{d}^k$.

We note that a line search that minimizes $f(\alpha)$ along the steepest-descent direction may not result in the lowest achievable function value over all search directions. This could happen, for example, when the current gradient $\nabla f(\boldsymbol{x}^k)$ points away from the local minimum, as is shown in the example presented at the end of the section.

A further weakness of the steepest descent method is that it becomes slow as the minimum is approached. This can be seen by examining the function derivative $f'(\alpha_k)$, which is computed as follows:

$$\frac{d}{d\alpha_k} f(\boldsymbol{x}^k + \alpha_k \boldsymbol{d}^k) = \nabla f(\boldsymbol{x}^{k+1})^T \boldsymbol{d}^k \tag{7.15}$$

The above result implies that the gradient $\nabla f(\boldsymbol{x}^{k+1})$ is normal to $\boldsymbol{d}^k$, i.e., in the case of steepest descent, normal to $\nabla f(\boldsymbol{x}^k)$. This implies a zigzag type progression towards the minimum that results in its slow progress. Due to its above weaknesses, the steepest descent method does not find much use in practice.

Rate of Convergence. The steepest-descent method displays linear convergence. In the case of quadratic functions, its rate constant is bounded by the following inequality (Griva, Nash & Sofer 2009, p. 406):

$$C = \frac{f(\boldsymbol{x}^{k+1}) - f(\boldsymbol{x}^*)}{f(\boldsymbol{x}^k) - f(\boldsymbol{x}^*)} \leq \left(\frac{cond(\boldsymbol{A}) - 1}{cond(\boldsymbol{A}) + 1}\right)^2 \tag{7.16}$$

The above result uses $f(\boldsymbol{x}^k) - f(\boldsymbol{x}^*)$, which converges at the same rate as $\left\|\boldsymbol{x}^k - \boldsymbol{x}^*\right\|$. Further, when using steepest-descent method with general nonlinear functions, the bound holds for $\boldsymbol{A} = \nabla^2 f(x^*)$.

Download free eBooks at bookboon.com

Preconditioning. As with all gradient methods, preconditioning aimed at reducing the condition number of the Hessian matrix can be employed to aid convergence of the steepest-descent method. To illustrate this point, we consider the cost function: $f(\boldsymbol{x}) = 0.1x_1^2 + x_2^2 = \boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x}, \boldsymbol{A} = diag\ (0,1,1)$, and define a linear transformation: $\boldsymbol{x} = \boldsymbol{P}\boldsymbol{y}$, where $\boldsymbol{P} = diag(\sqrt{10},\ 1)$. Then, the objective function is transformed as: $f(\boldsymbol{x}) = \boldsymbol{y}^T\boldsymbol{P}^T\boldsymbol{A}\boldsymbol{P}\boldsymbol{y}$, where the matrix product $\boldsymbol{P}^T\boldsymbol{A}\boldsymbol{P} = \boldsymbol{I}$ has a condition number of unity, indicating that the steepest-descent method will now converge in a single iteration.

An example of the steepest descent method is now presented.

Example 7.2: Steepest Descent

We consider minimizing $f(\boldsymbol{x}) = 0.1x_1^2 + x_2^2$ from an initial estimate $\boldsymbol{x}^0 = (5,1)$. The gradient of $f(\boldsymbol{x})$ is computed as , and $\nabla f(\boldsymbol{x}) = \begin{bmatrix} 0.2x_1 \\ 2x_2 \end{bmatrix}$, and $\nabla f(\boldsymbol{x}^0) = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$. Using the steepest-descent rule, the line search problem is given as: $\min_{\alpha} f(\alpha) = 0.1(5-\alpha)^2 + (1-2\alpha)^2$. The exact solution is found by setting $f'(\alpha) = 8.2\alpha - 5 = 0$, or $\alpha = 0.61$. Therefore, $\boldsymbol{x}^1 = \begin{bmatrix} 4.39 \\ -0.22 \end{bmatrix}$, and $f(\boldsymbol{x}^1) = 1.98$. Next, we try an arbitrary search direction $\boldsymbol{d}^0 = \begin{bmatrix} -1 \\ 0 \end{bmatrix}$, which gives $f(\alpha) = 0.1(5-\alpha)^2$+1, and a similar minimization results in $f'(\alpha) = 0.2\alpha - 1 = 0$, or $\alpha = 5$, for which, $\boldsymbol{x}^1 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$, and $f(\boldsymbol{x}^1) = 1$, which provides a better estimate of the actual minimum (0,0).

7.3.2 Conjugate-Gradient Methods

Conjugate-gradient (CG) methods employ conjugate vectors with respect to the Hessian matrix, as search directions in successive iterations; these directions hold the promise to minimize the function in n steps. The CG methods are popular in practice due to their low memory requirements and strong local and global convergence properties.

Let $\boldsymbol{d}^0, \boldsymbol{d}^2, \dots, \boldsymbol{d}^{n-1}$, where $\boldsymbol{d}^{i^T}\boldsymbol{A}\boldsymbol{d}^j = 0, i \neq j$, denote conjugate directions with respect to $\boldsymbol{A}$ matrix, and let $\boldsymbol{g}_k$ denote $\nabla f(\boldsymbol{x}^k)$. Then, starting from the steepest descent direction, we can use the following procedure to generate $\boldsymbol{A}$-conjugate directions:

$$\boldsymbol{d}^0 = -\boldsymbol{g}_0; \quad \boldsymbol{d}^{k+1} = -\boldsymbol{g}_{k+1} + \beta_k \boldsymbol{d}^k \qquad k \geq 0 \tag{7.17}$$

Next, application of the conjugacy condition results in:

$$\boldsymbol{d}^{k^T}\boldsymbol{A}\boldsymbol{d}^{k+1} = -\boldsymbol{d}^{k^T}\boldsymbol{A}g_{k+1} + \beta_k \boldsymbol{d}^{k^T}\boldsymbol{A}\boldsymbol{d}^k = 0, \text{or } \beta_k = \frac{\boldsymbol{g}_{k+1}^T\boldsymbol{A}\boldsymbol{d}^k}{\boldsymbol{d}^{k^T}\boldsymbol{A}\boldsymbol{d}^k} \tag{7.18}$$

Download free eBooks at bookboon.com

where we note that this expression can be further simplified if additional assumptions regarding the function and the line search algorithm are made. For example, since in the case of a quadratic function: $\boldsymbol{g}_{k+1} - \boldsymbol{g}_k = \boldsymbol{A}(\boldsymbol{x}_{k+1} - \boldsymbol{x}_k) = \alpha_k \boldsymbol{A}\boldsymbol{d}^k$, by substituting $\boldsymbol{A}\boldsymbol{d}^k = \frac{1}{\alpha_k}(\boldsymbol{g}_{k+1} - \boldsymbol{g}_k)$ in (7.18), we obtain: $\beta_k = \frac{\boldsymbol{g}_{k+1}^T(\boldsymbol{g}_{k+1} - \boldsymbol{g}_k)}{\boldsymbol{d}^{kT}(\boldsymbol{g}_{k+1} - \boldsymbol{g}_k)}$ (the Hestenes-Stiefel formula). Further, in the case of exact line search, $\boldsymbol{g}_{k+1}^T \boldsymbol{d}^k = 0$, $\beta_k = \frac{\boldsymbol{g}_{k+1}^T(\boldsymbol{g}_{k+1} - \boldsymbol{g}_k)}{\boldsymbol{g}_k^T \boldsymbol{g}_k}$ (the Polak-Ribiere formula). Finally, since $\boldsymbol{g}_{k+1}^T \boldsymbol{d}^k = \boldsymbol{g}_{k+1}^T(-\boldsymbol{g}_k + \beta_{k-1}\boldsymbol{d}^{k-1}) = 0$, ,whereas for quadratic functions, $\boldsymbol{g}_{k+1} = \boldsymbol{g}_k + \alpha_k \boldsymbol{A}\boldsymbol{d}^k$; therefore, by exact line search condition, $\boldsymbol{g}_{k+1}^T \boldsymbol{g}_k = \beta_{k-1}(\boldsymbol{g}_k + \alpha_k \boldsymbol{A}\boldsymbol{d}^k)^T \boldsymbol{d}^{k-1} = 0$, resulting in $\beta_k = \frac{\boldsymbol{g}_{k+1}^T \boldsymbol{g}_{k+1}}{\boldsymbol{g}_k^T \boldsymbol{g}_k}$ (the Fletcher-Reeves formula). Other versions of β_k have also been proposed.

The significance of the conjugacy property is apparent if we formulate a solution as: $y = \sum_{i=1}^{n} \alpha_i \boldsymbol{d}^i$, which is composed of n conjugate vectors. Then, the minimization problem is decomposed into a set of one-dimensional problems given as:

$$\min_y f(\boldsymbol{y}) = \sum_{i=1}^{n} \min_{\alpha_i} \left(\frac{1}{2} \alpha_i^2 \boldsymbol{d}^{iT} \boldsymbol{A}\boldsymbol{d}^i - \alpha_i \boldsymbol{c}^T \boldsymbol{d}^i \right) \tag{7.19}$$

Then, by setting the derivative with respect to α_i equal to zero, we obtain: $\alpha_i \boldsymbol{d}^{iT} \boldsymbol{A}\boldsymbol{d}^i - \boldsymbol{c}^T \boldsymbol{d}^i = 0$, leading to: $\alpha_i = \frac{\boldsymbol{c}^T \boldsymbol{d}^i}{\boldsymbol{d}^{iT} \boldsymbol{A}\boldsymbol{d}^i}$. The CG method iteratively determines conjugate directions $\boldsymbol{d}^i$ and their coefficients α_i.

Download free eBooks at bookboon.com

A Conjugate-gradient algorithm that uses residuals: $\boldsymbol{r}_i = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_i, i = 1,2, \dots, n,$ is given below:

Conjugate-Gradient Algorithm (Griva, Nash & Sofer, p454):

Init: Choose $\boldsymbol{x}_0 = \boldsymbol{0}, \boldsymbol{r}_0 = \boldsymbol{b}, \boldsymbol{d}^{(-1)} = 0, \beta_0 = 0.$

For $i = 0,1, \dots$

1. Check convergence: if $\|\boldsymbol{r}_i\| < \epsilon,$ stop.
2. If $i > 0$, set $\beta_i = \frac{\boldsymbol{r}_i^T \boldsymbol{r}_i}{\boldsymbol{r}_{i-1}^T \boldsymbol{r}_{i-1}}$
3. Set $\boldsymbol{d}^i = \boldsymbol{r}_i + \beta_i \boldsymbol{d}^{i-1}; \alpha_i = \frac{\boldsymbol{r}_i^T \boldsymbol{r}_i}{\boldsymbol{d}^{iT} \boldsymbol{A} \boldsymbol{d}^i}; \boldsymbol{x}_{i+1} = \boldsymbol{x}_i + \alpha_i \boldsymbol{d}^i; \boldsymbol{r}_{i+1} = \boldsymbol{r}_i - \alpha_i \boldsymbol{A} \boldsymbol{d}^i.$

Preconditioning. In all gradient-based methods, the convergence rates improve when the Hessian matrix has a low condition number. Preconditioning, or scaling, aimed at reducing the condition number, therefore, helps to speed up the convergence rates. Preconditioning involves a linear transformation: $\boldsymbol{x} = \boldsymbol{P}\boldsymbol{y},$ where $\boldsymbol{P}$ is invertible.

In the case of CG method, as a result of preconditioning, the conjugate directions are modified as:

$$\boldsymbol{d}^0 = -\boldsymbol{P}\boldsymbol{g}_0; \quad \boldsymbol{d}^{k+1} = -\boldsymbol{P}\boldsymbol{g}_{k+1} + \beta_k \boldsymbol{d}^k \qquad k \geq 0 \tag{7.20}$$

The modified CG parameter (in the case of Fletcher-Reeves formula) is given as: $\beta_k = \frac{\boldsymbol{g}_{k+1}^T \boldsymbol{P} \boldsymbol{g}_{k+1}}{\boldsymbol{g}_k^T \boldsymbol{P} \boldsymbol{g}_k}.$ Finally, the CG algorithm is modified to include preconditioning as follows:

Preconditioned Conjugate-Gradient Algorithm (Griva, Nash & Sofer, p. 475):

Initialize: Choose $\boldsymbol{x}_0 = \boldsymbol{0}, \boldsymbol{r}_0 = \boldsymbol{b}, \boldsymbol{d}^{(-1)} = 0, \beta_0 = 0.$
For $i = 0,1, \dots$

1. Check convergence: if $\|\boldsymbol{r}_i\| < \epsilon,$ stop.
2. Set $\boldsymbol{z}_i = \boldsymbol{P}^{-1} \boldsymbol{r}_i.$ If $i > 0$, set $\beta_i = \frac{\boldsymbol{r}_i^T \boldsymbol{z}_i}{\boldsymbol{r}_{i-1}^T \boldsymbol{z}_{i-1}}.$
3. Set $\boldsymbol{d}^i = \boldsymbol{z}_i + \beta_i \boldsymbol{d}^{i-1}; \alpha_i = \frac{\boldsymbol{r}_i^T \boldsymbol{z}_i}{\boldsymbol{d}^{iT} \boldsymbol{A} \boldsymbol{d}^i}; \boldsymbol{x}_{i+1} = \boldsymbol{x}_i + \alpha_i \boldsymbol{d}^i; \boldsymbol{r}_{i+1} = \boldsymbol{r}_i - \alpha_i \boldsymbol{A} \boldsymbol{d}^i.$

Download free eBooks at bookboon.com

Rate of Convergence. Conjugate gradient methods achieve superlinear convergence, which degenerates to linear convergence if the initial direction is not chosen as the steepest descent direction. In the case of quadratic functions, the minimum is reached exactly in n iterations. For general nonlinear functions, convergence in $2n$ iterations is to be expected. Nonlinear CG methods typically have the lowest per iteration computational costs.

An example of the CG method is given below.

Example 7.3: Conjugate-gradient method

We wish to solve the following minimization problem: $\min_{\boldsymbol{x}} f(x_1, x_2) = x_1^2 + 0.5x_2^2 - x_1x_2$, where: $\nabla f(\boldsymbol{x})^T = [2x_1 - x_2,\ x_2 - x_1]$.

Let $x_0 = (1,1)$, then: $\nabla f(\boldsymbol{x}^0) = \boldsymbol{c}^0 = [1,0]^T$, and we set $\boldsymbol{d}^0 = -\boldsymbol{c}^0 = [-1,0]^T$, which results in: $\boldsymbol{x}^1 = [1-\alpha, 1]^T$, and: $f(\alpha) = (1-\alpha)^2 + \alpha - 0.5$. Setting $f'(\alpha) = 0$, we obtain: $\alpha = 0.5$, and the solution estimate is updated as $\boldsymbol{x}^1 = [0.5, 1]^T$.

In the second iteration, we set $\boldsymbol{d}^1 = -\boldsymbol{c}^1 + \beta_0 \boldsymbol{d}^0$, where $\boldsymbol{c}^1 = [0, 0.5]^T, \beta_0 = \frac{\|\boldsymbol{c}^1\|}{\|\boldsymbol{c}^0\|} = 0.25$. Accordingly, $\boldsymbol{d}^1 = [-0.25, -0.5]^T, \boldsymbol{x}^2 = (1 - 0.5\alpha)[0.5, 1]^T$, and $f(\alpha) = 0.25(1 - 0.5\alpha)^2$. Again, by setting $f'(\alpha) = 0$, we obtain $\alpha = 2$, which gives $\boldsymbol{x}^2 = [0, 0]$. We note that the minimum of a quadratic function of two variables is reached in two iterations.

7.3.3 Newton's Method

Newton's method for finding the zero of a nonlinear function was earlier introduced in Section 2.11. Here we apply Newton's method to solve the nonlinear equation resulting from the application of FONC: $\nabla f(\boldsymbol{x}) = 0$. We use a linear approximation to $\nabla f(\boldsymbol{x})$ to apply this condition as:

$$\nabla f\left(\boldsymbol{x}^k + \boldsymbol{d}\right) \cong \nabla f\left(\boldsymbol{x}^k\right) + \nabla^2 f(x_k)\, \boldsymbol{d} = \boldsymbol{0} \tag{7.21}$$

Then, the direction vector is solved from a system of linear equations given as:

$$\nabla^2 f(\boldsymbol{x}_k)\boldsymbol{d} = -\nabla f(\boldsymbol{x}_k) \tag{7.22}$$

which leads to the following update formula:

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k - \left(\nabla^2 f(\boldsymbol{x}_k)\right)^{-1} \nabla f(\boldsymbol{x}_k) = \boldsymbol{x}_k - \boldsymbol{H}_k^{-1}\boldsymbol{g}_k \tag{7.23}$$

Download free eBooks at bookboon.com

We note that the above formula can also be obtained via second order Taylor series expansion of $f(\boldsymbol{x})$ given as:

$$f(\boldsymbol{x}^k + \boldsymbol{d}) = f(\boldsymbol{x}_k) + \nabla f(\boldsymbol{x}_k)^T \boldsymbol{d} + \frac{1}{2}\boldsymbol{d}^T \boldsymbol{H}_k \boldsymbol{d} = q_k(\boldsymbol{d}) \tag{7.24}$$

The above expression implies that at every iteration Newton's method approximates $f(\boldsymbol{x})$ by a quadratic function: $q_k(\boldsymbol{d})$; it then solves the minimization problem: $\min_d q_k(\boldsymbol{d})$ and updates the current estimate as: $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \boldsymbol{d}$. Further, the above solution assumes that $q_k(\boldsymbol{d})$ is convex, i.e., $\boldsymbol{H}_k = \nabla^2 f(\boldsymbol{x}_k)$ is positive-definite.

The application of Newton's method relies on the positive-definite assumption for $\boldsymbol{H}_k = \nabla^2 f(\boldsymbol{x}_k)$. If $\nabla^2 f(\boldsymbol{x}_k)$ is positive-definite, then a factorization of the form: $\nabla^2 f(\boldsymbol{x}_k) = \boldsymbol{L}\boldsymbol{D}\boldsymbol{L}^T$, where $d_{ii} > 0$, can be used to solve for the resulting system of linear equations: $(\boldsymbol{L}\boldsymbol{D}\boldsymbol{L}^T)\boldsymbol{d} = -\nabla f(\boldsymbol{x}_k)$. If at any point $\boldsymbol{D}$ is found to have negative entries, i.e., if $d_{ii} \leq 0$, then it should be replaced by a positive value, such as $|d_{ii}|$. This correction amounts to adding a diagonal matrix $\boldsymbol{E}$, such that $\nabla^2 f(\boldsymbol{x}_k) + \boldsymbol{E}$ is positive-definite.

An algorithm for Newton's method is given below.

Download free eBooks at bookboon.com

Newton's Method (Griva, Nash, & Sofer, p. 373):

Initialize: Choose $\boldsymbol{x}_0$, specify ϵ
For $k = 0,1, \ldots$

1. Check convergence: If $\|\nabla f(\boldsymbol{x}_k)\| < \epsilon$, stop
2. Factorize modified Hessian as $\nabla^2 f(\boldsymbol{x}_k) + \boldsymbol{E} = \boldsymbol{LDL}^T$ and solve $(\boldsymbol{LDL}^T)\boldsymbol{d} = -\nabla f(\boldsymbol{x}_k)$ for $\boldsymbol{d}$
3. Perform line search to determine α_k and update the solution estimate as $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \alpha_k \boldsymbol{d}^k$

Rate of Convergence. Newton's method achieves quadratic rate of convergence in the close neighborhood of the optimal point, and superlinear rate of convergence otherwise. Moreover, due to its high computational and storage costs, classic Newton's method is rarely used in practice.

7.3.4 Quasi-Newton Methods

Quasi-Newton methods that use low-cost approximations to the Hessian matrix are the among most widely used methods for nonlinear problems. These methods represent a generalization of one-dimensional secant method, which approximates the second derivative as: $f''(x_k) \cong \frac{f'(x_k) - f'(x_{k-1})}{x_k - x_{k-1}}$.

In the multi-dimensional case, the secant method translates into the following:

$$\nabla^2 f(\boldsymbol{x}_k)(\boldsymbol{x}_k - \boldsymbol{x}_{k-1}) \cong \nabla f(\boldsymbol{x}_k) - \nabla f(\boldsymbol{x}_{k-1}) \tag{7.25}$$

Thus, if the Hessian is approximated by a positive-definite matrix $\boldsymbol{H}_k$, then $\boldsymbol{H}_k$ then is required to satisfy the following secant condition:

$$\boldsymbol{H}_k(\boldsymbol{x}_k - \boldsymbol{x}_{k-1}) = \nabla f(\boldsymbol{x}_k) - \nabla f(\boldsymbol{x}_{k-1}) \tag{7.26}$$

Whereas, the above condition places n constraints on the structure of $\boldsymbol{H}_k$, further constraints may be added to completely specify $\boldsymbol{H}_k$ as well as to preserve its symmetry.

The quasi-Newton methods aim to iteratively update $\boldsymbol{H}_k$ via:

1. The direct update: $\boldsymbol{H}_{k+1} = \boldsymbol{H}_k + \Delta \boldsymbol{H}_k, \ \boldsymbol{H}_0 = \boldsymbol{I}$; or
2. The inverse update: $\boldsymbol{F}_{k+1} = \boldsymbol{F}_k + \Delta \boldsymbol{F}_k, \ \boldsymbol{F} = \boldsymbol{H}^{-1}, \ \boldsymbol{F}_0 = \boldsymbol{I}$.

Once $\boldsymbol{H}_k$ is available, it can be employed to solve for the current search direction from: $\boldsymbol{H}_k \boldsymbol{d} = -\nabla f(\boldsymbol{x}_k)$ or from: $\boldsymbol{d} = -\boldsymbol{F}_k \nabla f(\boldsymbol{x}_k)$.

Download free eBooks at bookboon.com

To proceed further, let $\boldsymbol{s}_k = \boldsymbol{x}_k - \boldsymbol{x}_{k-1}$, $\boldsymbol{y}_k = \nabla f(\boldsymbol{x}_k) - \nabla f(\boldsymbol{x}_{k-1})$; then, a symmetric rank-one update formula for $\boldsymbol{H}_k$ is given as (Griva, Nash & Sofer, p.414):

$$\boldsymbol{H}_{k+1} = \boldsymbol{H}_k + \frac{(\boldsymbol{y}_k - \boldsymbol{H}_k\boldsymbol{s}_k)(\boldsymbol{y}_k - \boldsymbol{H}_k\boldsymbol{s}_k)^T}{(\boldsymbol{y}_k - \boldsymbol{H}_k\boldsymbol{s}_k)^T\boldsymbol{s}_k} \tag{7.27}$$

However, the above formula, while obeying the secant condition, $\boldsymbol{H}_{k+1}\boldsymbol{s}_k = \boldsymbol{y}_k$, does not ensure that $\boldsymbol{H}_k$ is positive-definite. Next, a class of symmetric rank-two update formulas that ensures positive-definiteness of $\boldsymbol{H}_k$ are defined by:

$$\boldsymbol{H}_{k+1} = \boldsymbol{H}_k - \frac{(\boldsymbol{H}_k\boldsymbol{s}_k)(\boldsymbol{H}_k\boldsymbol{s}_k)^T}{\boldsymbol{s}_k{}^T\boldsymbol{H}_k\boldsymbol{s}_k} + \frac{\boldsymbol{y}_k\boldsymbol{y}_k{}^T}{\boldsymbol{y}_k{}^T\boldsymbol{s}_k} + \phi(\boldsymbol{s}_k{}^T\boldsymbol{H}_k\boldsymbol{s}_k)\boldsymbol{v}_k\boldsymbol{v}_k{}^T \tag{7.28}$$

where $\boldsymbol{v}_k = \frac{\boldsymbol{y}_k}{\boldsymbol{y}_k{}^T\boldsymbol{s}_k} - \frac{\boldsymbol{H}_k\boldsymbol{s}_k}{\boldsymbol{s}_k{}^T\boldsymbol{H}_k\boldsymbol{s}_k}$ and $\phi \in [0,1)$. Two popular choices for ϕ are: $\phi = 0$ and $\phi = 1$, resulting in the well-known DFP (Davison, Fletcher, and Powell) and BGFS (Broyden, Fletcher, Goldfarb, and Shanno) update formulas.

The former (DFP formula) results in the following inverse Hessian update:

$$\boldsymbol{F}_{k+1} = \boldsymbol{F}_k - \frac{(\boldsymbol{F}_k\boldsymbol{y}_k)(\boldsymbol{F}_k\boldsymbol{y}_k)^T}{\boldsymbol{y}_k{}^T\boldsymbol{F}_k\boldsymbol{y}_k} + \frac{\boldsymbol{s}_k\boldsymbol{s}_k{}^T}{\boldsymbol{y}_k{}^T\boldsymbol{s}_k} \tag{7.29}$$

The latter (BFGS formula) results in a direct Hessian update given as:

$$\boldsymbol{H}_{k+1} = \boldsymbol{H}_k - \frac{(\boldsymbol{H}_k\boldsymbol{s}_k)(\boldsymbol{H}_k\boldsymbol{s}_k)^T}{\boldsymbol{s}_k{}^T\boldsymbol{H}_k\boldsymbol{s}_k} + \frac{\boldsymbol{y}_k\boldsymbol{y}_k{}^T}{\boldsymbol{y}_k{}^T\boldsymbol{s}_k} \tag{7.30}$$

We note that the Hessian in the case of a quadratic function, $q(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^T\boldsymbol{Q}\boldsymbol{x} - \boldsymbol{c}^T\boldsymbol{x}$, obeys the secant condition, $\boldsymbol{Q}\boldsymbol{s}_k = \boldsymbol{y}_k$, which shows that a symmetric positive-definite $\boldsymbol{H}_k$ in a quasi-Newton method locally approximates the quadratic behavior.

The quasi-Newton algorithm is given below.

Quasi-Newton Algorithm (Griva, Nash & Sofer, p. 415):
Initialize: Choose $\boldsymbol{x}_0$, $\boldsymbol{H}_0$ (e.g., $\boldsymbol{H}_0 = \boldsymbol{I}$), specify ε
For $k = 0,1, \ldots$

1. Check convergence: If $\|\nabla f(\boldsymbol{x}_k)\| < \varepsilon$, stop
2. Solve $\boldsymbol{H}_k\boldsymbol{d} = -\nabla f(\boldsymbol{x}_k)$ for $\boldsymbol{d}^k$
3. Solve for $\min_{\alpha} f\left(\boldsymbol{x}^k + \alpha\boldsymbol{d}^k\right)$ for α_k, and update $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \alpha_k\,\boldsymbol{d}^k$
4. Compute $\boldsymbol{s}_k, \boldsymbol{y}_k$, and update $\boldsymbol{H}_k$ as per (5.19) or (5.20)

Download free eBooks at bookboon.com

Rate of Convergence. Quasi-Newton methods achieve superlinear convergence, thus rivaling the second order methods for solving nonlinear programming (NP) problems.

Example 7.4: Quasi-Newton method

As an example, we consider the following NL problem:
$\min_{x_1,x_2} f(x_1,x_2) = 2x_1^2 - x_1x_2 + x_2^2$
where $\nabla f = \begin{bmatrix} 4x_1 - x_2 \\ 2x_2 - x_1 \end{bmatrix}$, $H = \begin{bmatrix} 4 & -1 \\ -1 & 2 \end{bmatrix}$.

Let $x^0 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, H_0 = I$; then, $f^0 = 2,\ d^0 = -\nabla f(x^0) = \begin{bmatrix} -3 \\ -1 \end{bmatrix}$. Using: $f(\alpha) = 2(1-3\alpha)^2 + (1-\alpha)^2 - (1-3\alpha)(1-\alpha)$. and putting $f'(\alpha) = 0$ gives $\alpha = \frac{5}{16}$. Then, $s_1 = \alpha d^0 = \frac{5}{16}\begin{bmatrix} 1 \\ 1 \end{bmatrix},\ g^1 = y_1, \boldsymbol{x}^1 = \left(\frac{3}{4}, \frac{3}{4}\right)$.

For the Hessian update, we have: $f^1 = 0.5625,\ g_1 = -0.125,\ g_2 = g_3 = -0.75; \boldsymbol{c}^1 = [0.75, 0.75]$; and, for $\alpha = 0.25,\ \boldsymbol{s}^0 = [-0.25, -0.25] = \boldsymbol{z}^0 = \boldsymbol{y}^0,\ \xi_1 = \xi_2 = 0.125,\ \theta = 1,\ \boldsymbol{w}^0 = \boldsymbol{y}^0,\ \xi_3 = \xi_1$; then, the Hessian update is computed as: $\boldsymbol{D}^0 = 8\begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix},\ \boldsymbol{E}^0 = 8\begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix},\ \boldsymbol{H}^1 = \boldsymbol{H}^0$.

We next proceed to discuss the trust-region methods of solving NP problems.

Download free eBooks at bookboon.com

7.3.5 Trust-Region Methods

Trust-region methods locally employ a quadratic approximation $q_k(\boldsymbol{x}_k)$ to the nonlinear objective function; they were originally proposed to solve the nonlinear least-squares problems, but have since been adapted to solve more general optimization problems.

The quadratic approximation is given as: $q(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^T\boldsymbol{Q}\boldsymbol{x} - \boldsymbol{c}^T\boldsymbol{x}$, and is valid in a limited neighborhood $\Omega_k = \{\boldsymbol{x}: \|\boldsymbol{\Gamma}(\boldsymbol{x} - \boldsymbol{x}_k)\| \leq \Delta_k\}$ of $\boldsymbol{x}_k$, where $\boldsymbol{\Gamma}$ is a scaling parameter. The method then aims to find a $\boldsymbol{x}_{k+1} \in \Omega_k$, which results in sufficient decrease in $f(\boldsymbol{x})$. At each iteration k, trust-region algorithm solves a constrained optimization sub-problem defined by**:**

$$\min_{\boldsymbol{d}} q_k(\boldsymbol{d}) = f(\boldsymbol{x}_k) + \nabla f(\boldsymbol{x}_k)^T\boldsymbol{d} + \frac{1}{2}\boldsymbol{d}^T\nabla^2 f(\boldsymbol{x}_k)\boldsymbol{d} \tag{7.31}$$
subject to $\|\boldsymbol{d}\| \leq \Delta_k$

Using a Lagrangian function approach the first order optimality conditions are given as:

$$(\nabla^2 f(\boldsymbol{x}_k) + \lambda\boldsymbol{I})\boldsymbol{d}^k = -\nabla f(\boldsymbol{x}_k) \tag{7.32}$$

where $\lambda \geq 0$ is the Lagrange multiplier associated with the constraint, and $(\nabla^2 f(\boldsymbol{x}_k) + \lambda\boldsymbol{I})$ is a positive-definite matrix. The quality of the quadratic approximation is estimated by: $\gamma_k = \frac{f(x_k) - f(x_{k+1})}{q_k(x_k) - q_k(x_{k+1})}$. . If this ratio is close to unity, the trust region may be expanded in the next iteration.

The resulting search direction $\boldsymbol{d}^k$ is a function of Lagrange multiplier λ: $\boldsymbol{d}^k = \boldsymbol{d}^k(\lambda)$. Thus, for $\lambda = 0$, a sufficiently large Δ_k, and for a positive-definite $\nabla^2 f(\boldsymbol{x}_k)$, $\boldsymbol{d}^k(0)$ reduces to the Newton's direction. Whereas, for $\Delta_k = 0$, $\lambda \to \infty$, and $\boldsymbol{d}^k(\lambda)$ and aligns with the steepest-descent direction.

The trust-region algorithm is given as follows:

Trust-Region Algorithm (Griva, Nash & Sofer, p.392):

Initialize: Choose $\boldsymbol{x}_0$, Δ_0; specify $\varepsilon, 0 < \mu < \eta < 1$ (e.g., $\mu = \frac{1}{4}$; $\eta = \frac{3}{4}$)

For $k = 0,1, \ldots$

1. Check convergence: If $\|\nabla f(\boldsymbol{x}_k)\| < \varepsilon$, stop
2. Solve $\min_{\boldsymbol{d}} q_k(\boldsymbol{d})$ subject to $\|\boldsymbol{d}\| \leq \Delta_k$
3. Compute γ_k,
   a) if $\gamma_k < \mu$, set $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k$, $\Delta_{k+1} = \frac{1}{2}\Delta_k$
   b) else if $\gamma_k < \eta$, set $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \boldsymbol{d}^k$, $\Delta_{k+1} = \Delta_k$
   c) else set $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \boldsymbol{d}^k$, $\Delta_{k+1} = 2\Delta_k$

Download free eBooks at bookboon.com

7.4 Computer Methods for Solving the Constrained Problems

In this section, we describe the numerical methods devised for solving constrained nonlinear optimization problems. These methods fall into two broad categories: the first category includes penalty, barrier, and augmented Lagrangian methods that are an extension of the methods developed for unconstrained problems, and are collectively known as the transformation methods. The second category includes methods that iteratively approximate the nonlinear problem as a series of LP or QP problems and use the LP or QP methods to solve it.

In order to discuss these methods, we consider a general optimization problem described as:

$$\min_{\boldsymbol{x}} \ f(\boldsymbol{x})$$
$$\text{Subject to } \begin{cases} h_i(\boldsymbol{x}) = 0, \ i = 1, \dots, p; \\ g_j(\boldsymbol{x}) \le 0, j = i, \dots, m; \\ x_{iL} \le x_i \le x_{iU}, \ i = 1, \dots, n. \end{cases} \tag{7.33}$$

Prominent computer methods for solving constrained optimization problems are described in this and the following section.

7.4.1 Penalty and Barrier Methods

The Penalty and Barrier methods are extensions of the numerical methods developed for solving unconstrained optimization problems. Both methods employ a composite of objective and constraint functions where the constraints are assigned a high violation penalty. Once a composite function has been defined for a set of penalty parameters, it can be minimized using any of the unconstrained optimization techniques. The penalty parameters can then be adjusted in successive iterations.

The Penalty and Barrier methods fall under sequential unconstrained minimization techniques (SUMTs). Because of their simplicity, SUMTs have been extensively developed and used in engineering design problems. The SUMTs generally employ a composite function of the following form (Arora, p. 477):

$$\Phi(\boldsymbol{x}, r) = f(\boldsymbol{x}) + P(g(\boldsymbol{x}), h(\boldsymbol{x}), \boldsymbol{r}) \tag{7.34}$$

where $g(\boldsymbol{x})$ and $h(\boldsymbol{x})$ are, respectively, the inequality and equality constraints, and $\boldsymbol{r}$ is a vector of penalty parameters. Depending on their region of iteration, these methods are further divided into Penalty or Barrier methods as described below:

Penalty Function Method. A penalty function method that iterates through the infeasible region of space, employs a quadratic loss function of the following form:

$$P(g(\boldsymbol{x}), h(\boldsymbol{x}), \boldsymbol{r}) = r\left(\sum_i \left(g_i^+(\boldsymbol{x})\right)^2 + \sum_i \left(h_i(\boldsymbol{x})\right)^2\right); \ g_i^+(\boldsymbol{x}) = \max\left(0, g_i(\boldsymbol{x})\right), r > 0 \tag{7.35}$$

Download free eBooks at bookboon.com

Barrier Function Method. A barrier method that iterates through the feasible region of space, and is only applicable to inequality constrained problems, employs a log barrier function of the following form:

$$P(g(\boldsymbol{x}), h(\boldsymbol{x}), \boldsymbol{r}) = \frac{1}{r}\sum_i \log\left(-g_i(x)\right) \tag{7.36}$$

For both penalty and barrier methods, convergence implies that as $r \to \infty$, $\boldsymbol{x}(r) \to \boldsymbol{x}^*$, where $\boldsymbol{x}(r)$ minimizes $\Phi(\boldsymbol{x}, r)$. To improve convergence, *r* may be replaced by a sequence $\{r^k\}$. We, however, note that since the Hessian of the unconstrained function becomes ill-conditioned for large *r*, both methods are ill-behaved near the constraint boundary.

7.4.2 The Augmented Lagrangian Method

As an alternative to the penalty and barrier methods described above, the augmented Lagrangian (AL) methods add a quadratic penalty term to the Lagrangian function that also includes multipliers for penalizing individual constraint violations. The resulting AL method is generally more effective than penalty and barrier methods, and is commonly employed to solve Finite Element Analysis problems.

The augmented Lagrangian method is introduced below using an equality constrained optimization problem where the problem is given as (Belegundu and Chandrupatla, p. 276):

$$\begin{aligned} &\min_{\boldsymbol{x}} \ f(\boldsymbol{x}) \\ &\text{Subject to: } h_i(\boldsymbol{x}) = 0, \ i = 1, \dots, l \end{aligned} \tag{7.37}$$

Download free eBooks at bookboon.com

The augmented Lagrangian function for the problem is defined as:

$$\mathcal{P}(\boldsymbol{x},\boldsymbol{v},r) = f(\boldsymbol{x}) + \sum_j \left(v_j h_j(\boldsymbol{x}) + \frac{1}{2} r h_j^2(\boldsymbol{x}) \right) \tag{7.38}$$

In the above, v_j are the Lagrange multipliers and the additional term defines an exterior penalty function with r as the penalty parameter. The gradient and Hessian of the AL are computed as:

$$\begin{aligned} \nabla\mathcal{P}(\boldsymbol{x},\boldsymbol{v},r) &= \nabla f(\boldsymbol{x}) + \sum_j \Big(v_j + r h_j(\boldsymbol{x}) \Big) \nabla h_j(\boldsymbol{x}) \\ \nabla^2\mathcal{P}(\boldsymbol{x},\boldsymbol{v},r) &= \nabla^2 f(\boldsymbol{x}) + \sum_j \left(\Big(v_j + r h_j(\boldsymbol{x}) \Big) \nabla^2 h_j(\boldsymbol{x}) + r \nabla h_j^{\mathrm{T}} \nabla h_j(\boldsymbol{x}) \right) \end{aligned} \tag{7.39}$$

While the Hessian of the Lagrangian may not be uniformly positive definite, a large enough value of r makes the Hessian of AL positive definite at $\boldsymbol{x}$.

Next, since the AL is stationary at the optimum, then, paralleling the developments in the duality theory (Sec. 5.7), we can solve the above optimization problem via a min-max framework as follows: first, for a given r and $\boldsymbol{v}$, we define a dual function via the following minimization problem:

$$\psi(\boldsymbol{v}) = \min_{\boldsymbol{x}} \mathcal{P}(\boldsymbol{x},\boldsymbol{v},r) = f(\boldsymbol{x}) + \sum_j \left(v_j h_j(\boldsymbol{x}) + \frac{1}{2} r \Big(h_j(\boldsymbol{x}) \Big)^2 \right) \tag{7.40}$$

This step is then followed by a maximization problem defined as: $\max_{\boldsymbol{v}} \psi(\boldsymbol{v})$. The derivative of the dual function is computed as: $\frac{d\psi}{dv_j} = h_j(\boldsymbol{x}) + \nabla\psi^T \frac{d\boldsymbol{x}}{dv_j}$, where the latter term is zero, since $\nabla\psi = \nabla\mathcal{P} = 0$. Further, an expression for the Hessian is given as: $\frac{d^2\psi}{dv_i dv_j} = \nabla {h_i}^T \frac{d\boldsymbol{x}}{dv_j}$, where the $\frac{d\boldsymbol{x}}{dv_j}$ where the term can be obtained by differentiating $\nabla\psi = 0$, which gives: $\nabla h_j + \nabla^2\mathcal{P}\left(\frac{d\boldsymbol{x}}{dv_j}\right) = 0$, or $\nabla^2\mathcal{P}\left(\frac{d\boldsymbol{x}}{dv_j}\right) = -\nabla h_j$. Therefore, the Hessian is computed as:

$$\frac{d^2\psi}{dv_i dv_j} = -\nabla {h_i}^T (\nabla^2\mathcal{P})^{-1} \nabla h_j \tag{7.41}$$

The AL method proceeds as follows: we choose a suitable $\dot{v}$ and solve the minimization problem in (7.40) to define $\psi(v)$. We then solve the maximization problem to find the solution that minimizes the AL. The latter step can be done using gradient-based methods. For example, the Newton update for the maximization problem is given as:

$$\boldsymbol{v}^{k+1} = \boldsymbol{v}^k - \left(\frac{d^2\psi}{dv_i dv_j} \right)^{-1} \boldsymbol{h} \tag{7.42}$$

Download free eBooks at bookboon.com

For large $\boldsymbol{r}$, the update may be approximated as: $v_j^{k+1} = v_j^k + r_j h_j,\ j = 1, \dots, l$ (Belegundu and Chandrupatla, p. 278). For inequality constrained problems, the AL may be defined as (Arora, p. 480):

$$\mathcal{P}(\boldsymbol{x}, \boldsymbol{u}, r) = f(\boldsymbol{x}) + \sum_i \begin{cases} u_i g_i(\boldsymbol{x}) + \frac{1}{2} r g_i^2(\boldsymbol{x}), & \text{if } g_j + \frac{u_j}{r} \geq 0 \\ -\frac{1}{2r} u_i^2, & \text{if } g_j + \frac{u_j}{r} < 0 \end{cases} \tag{7.43}$$

The AL algorithm is given below.

The Augmented Lagrangian Algorithm (Arora, p. 480)

Initialize: estimate $x^0, u^0 \geq 0, v^0, r > 0$; choose $\alpha > 0, \beta > 1, \epsilon > 0, \kappa > 0, K = \infty$
For $k = 1,2, \dots$

1. Solve $\boldsymbol{x}^k = \min_{\boldsymbol{x}} \mathcal{P}(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{v}, r_k)$
2. Evaluate $h_i(\boldsymbol{x}^k), i = 1, .., l; g_j(\boldsymbol{x}^k), j = 1, \dots, m;$
   compute $\overline{K} = max\left\{|h_i|, i = 1, \dots, l; \max\left(g_j, -\frac{u_j}{r_k}\right), j = 1, \dots, m\right\}$
3. Check termination: If $\overline{K} \leq \kappa$ and $\left\|\nabla \mathcal{P}(\boldsymbol{x}^k)\right\| \leq \epsilon\ max\{1, \left\|\boldsymbol{x}^k\right\|\}$, quit
4. If $\overline{K} < K$ (i.e., constraint violations have improved), set $K = \overline{K}$
   Set $v_i^{k+1} = v_i^k + r_k h_i(\boldsymbol{x}^k); i = 1, \dots, l$. Set $u_j^{k+1} = u_j^k + r_k max\left\{g_j(\boldsymbol{x}^k), -\frac{u_j^k}{r_k}\right\}; j = 1, \dots, m.$
   If $\overline{K} > \frac{K}{\alpha}$, (i.e., constraint violations did not improve by a factor α), set $r_{k+1} = \beta r_k$

An example for the AL method is now presented.

Example 7.5: Design of cylindrical water tank (Belegundu and Chandrupatla, p. 278)

We consider the design of an open-top cylindrical water tank. We wish to maximize the volume of the tank for a given surface area A_0. Let d be the diameter and h be the height; then, the optimization problem is formulated as:

$$\max_{d,l} f(d, l) = \frac{\pi d^2 l}{4}$$

subject to $h: \frac{\pi d^2}{4} + \pi d l - A_0 = 0$

Download free eBooks at bookboon.com

We drop the constant $\frac{\pi}{4}$, convert to a minimization problem, assume $\frac{4A_0}{\pi} = 1$, and redefine the problem as:

$$\min_{d,l} \bar{f}(d,l) = -d^2 l$$
subject to h: $d^2 + 4dl - 1 = 0$

A Lagrangian function for the problem is formulated as: $\mathcal{L}(d,l,\lambda) = -d^2 l + \lambda(d^2 + 4dl - 1)$
The FONC for the problem are: $-2dl + 2\lambda(d + 2l) = 0, -d^2 + 4d\lambda = 0, d^2 + 4dl - 1 = 0$.
Using FONC, the optimal solution is given as: $d^* = 2l^* = 4\lambda^* = \frac{1}{\sqrt{3}}$.
The Hessian at the optimum point is given as: $\nabla^2 \mathcal{L}(d^*, l^*, \lambda^*) = \begin{bmatrix} -2\lambda & -4\lambda \\ -4\lambda & 0 \end{bmatrix}$. It is evident that the Hessian is not positive definite.

Next, the AL for the problem is formed as: $\mathcal{P}(d,l,\lambda,r) = -d^2 l + \lambda(d^2 + 4dl - 1) + \frac{1}{2} r(d^2 + 4dl - 1)^2$
The dual function is defined as: $\psi(\lambda) = \min_{d,l} \mathcal{P}(d,l,\lambda,r)$.
The dual optimization problem is then formulated as: $\max_{d,l} \psi(\lambda)$.
A plot of $\psi(\lambda)$ vs. λ vs. shows a concave function with $\lambda^* = \lambda_{max} = 0.144$.
The optimum values for the design variables are the same as above: $d^* = 2l^* = 0.577$.

Download free eBooks at bookboon.com

7.5 Sequential Linear Programming

The sequential linear programming (SLP) method aims to sequentially solve the nonlinear optimization problem as a series of linear programs. In particular, we employ the first order Taylor series expansion to iteratively develop and solve a new LP subprogram to solve the KKT conditions associated with the NP problem. SLP methods are generally not robust, and have been mostly replaced by SQP methods.

To develop the SLP method, let $\boldsymbol{x}^k$ denote the current estimate of design variables and let $\boldsymbol{d}$ denote the change in variable; then, we express the first order expansion of the objective and constraint functions in the neighborhood of $\boldsymbol{x}^k$ as:

$$\begin{aligned} f(\boldsymbol{x}^k + \boldsymbol{d}) &= f(\boldsymbol{x}^k) + \nabla f(\boldsymbol{x}^k)^T \boldsymbol{d} \\ g_i(\boldsymbol{x}^k + \boldsymbol{d}) &= g_i(\boldsymbol{x}^k) + \nabla g_i(\boldsymbol{x}^k)^T \boldsymbol{d},\ i = 1, \dots, m \\ h_j(\boldsymbol{x}^k + \boldsymbol{d}) &= h_j(\boldsymbol{x}^k) + \nabla h_j(\boldsymbol{x}^k)^T \boldsymbol{d},\ j = 1, \dots, l \end{aligned} \tag{7.44}$$

To proceed further, let: $f^k = f(\boldsymbol{x}^k),\ g_i^k = g_i(\boldsymbol{x}^k),\ h_j^k = h_j(\boldsymbol{x}^k)$; and define: $b_i = -g_i^k,\ e_j = -h_j^k,$ $\boldsymbol{c} = \nabla f(\boldsymbol{x}^k),\ \boldsymbol{a}_i = \nabla g_i(\boldsymbol{x}^k),\ \boldsymbol{n}_j = \nabla h_j(\boldsymbol{x}^k),\ \boldsymbol{A} = [\boldsymbol{a}_1, \boldsymbol{a}_2, \dots, \boldsymbol{a}_m],\ \boldsymbol{N} = [\boldsymbol{n}_1, \boldsymbol{n}_2, \dots, \boldsymbol{n}_l]$. Then, after dropping the constant term f^k from the objective function, we define the following LP subprogram for the current iteration of the NP problem (Arora, p. 498):

$$\begin{aligned} &\min_{\boldsymbol{d}} \bar{f} = \boldsymbol{c}^T \boldsymbol{d} \\ &\text{Subject to: } \boldsymbol{A}^T \boldsymbol{d} \le \boldsymbol{b},\ \boldsymbol{N}^T \boldsymbol{d} = \boldsymbol{e} \end{aligned} \tag{7.45}$$

where $\bar{f}$ represents the linearized change in the original cost function and the columns of $\boldsymbol{A}$ and $\boldsymbol{N}$ represent, respectively, the gradients of inequality and equality constraints. Since the objective and constraint functions are now linear, the resulting LP subproblem can be converted to standard form and solved via the Simplex method. Problems with a small number of variables can also be solved graphically or by application of KKT conditions to the LP problem.

The following points regarding the SLP method should be noted:

1. Since both positive and negative changes to design variables $\boldsymbol{x}^k$ are allowed, the variables d_i are unrestricted in sign and, therefore, must be replaced by $d_i = d_i^+ - d_i^-$ in the Simplex algorithm.
2. In order to apply the simplex method to the problem, the rhs parameters b_i, e_j are assumed non-negative, or else, the respective constraint must be multiplied with -1.
3. SLP methods require additional constraints of the form, $-\Delta_{il}^k \le d_i^k \le \Delta_{iu}^k$, termed as move limits, to bind the LP solution. These move limits represent the maximum allowed change in d_i in the current iteration. They are generally selected as a percentage (1–100%) of the design variable values. They serve dual purpose of binding the LP solution and obviating the need for line search in the current iteration. Restrictive move limits tend to make the SLP problem infeasible.

Download free eBooks at bookboon.com

The SLP algorithm is presented below:

SLP Algorithm (Arora, p. 508)**:**

Initialize: choose $\boldsymbol{x}^0,\ \varepsilon_1 > 0, \varepsilon_2 > 0$.
For $k = 0,1,2, \ldots$

1. Choose move limits $\Delta_{il}^k, \Delta_{iu}^k$ as some fraction of current design $\boldsymbol{x}^k$
2. Compute $f^k, \boldsymbol{c},\ g_i^k, h_j^k, b_i, e_j$
3. Formulate and solve the LP subproblem for $\boldsymbol{d}^k$
4. If and $g_i \le \varepsilon_1; i = 1, \ldots, m;\ |h_j| \le \varepsilon_1; i = 1, \ldots, p;$ and $\|\boldsymbol{d}^k\| \le \varepsilon_2$, stop
5. Substitute $\boldsymbol{x}^{k+1} \leftarrow \boldsymbol{x}^k + \alpha \boldsymbol{d}^k,\ \ k \leftarrow k + 1$.

The SLP algorithm is simple to apply, but should be used with caution in engineering design problems as it can easily run into convergence problems. The selection of move limits is one of trial and error and can be best achieved in an interactive mode.

An example is presented to explain the SLP method:

Example 7.6: Sequential Linear Programming

We perform one iteration of the SLP algorithm for the following NLP problem:

$$\min_{x_1, x_2} f(x_1, x_2) = x_1^2 - x_1 x_2 + x_2^2$$
Subject to: $1 - x_1^2 - x_2^2 \le 0;\ -x_1 \le 0, -x_2 \le 0$

The NLP problem is convex and has a single minimum at $\boldsymbol{x}^* = \left(\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}\right)$. The objective and constraint gradients are: $\nabla f^T = [2x_1 - x_2, 2x_2 - x_1],\ \nabla g_1^T = [-2x_1, -2x_2],\ \nabla g_2^T = [-1,0],\ \nabla g_3^T = [0, -1]$.

Let $\boldsymbol{x}^0 = (1,1),$ so that $f^0 = 1, \boldsymbol{c}^T = [1 \ \ 1]$; further, let $\varepsilon_1 = \varepsilon_2 = 0.001$; then, using SLP method, the resulting LP problem at the current step is defined as:

$$\min_{d_1, d_2} f(x_1, x_2) = d_1 + d_2$$
Subject to: $\begin{bmatrix} -2 & -2 \\ -1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} d_1 \\ d_2 \end{bmatrix} \le \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$

Download free eBooks at bookboon.com

Since the LP problem is unbounded, we may use 50% move limits to bind the solution. The resulting update is given as: $\boldsymbol{d}^* = \left[-\frac{1}{2},\ -\frac{1}{2}\right]^T$, so that $\boldsymbol{x}^1 = \left[\frac{1}{2},\ \frac{1}{2}\right]^T$, with resulting constraint violations given as: $g_i = \left\{\frac{1}{2}, 0,\ 0\right\}$. We note that smaller move limits in this step could have avoided resulting constraint violation.

The SLP algorithm is not robust as move limits need to be imposed to force a solution. In the following, a sequential quadratic problem that obviates the need for move limits is formulated and solved.

7.6 Sequential Quadratic Programming

The sequential quadratic programming (SQP) method improves on the SLP method by discarding the move limits in favor of more robust ways of binding the solution. Specifically, SQP adds $\|\boldsymbol{d}\|$, where $\boldsymbol{d}$ where represents the search direction, to the linear objective function (7.45) to define the resulting QP subproblem as follows (Arora, p. 514):

$$\min_{\boldsymbol{d}} \bar{f} = \boldsymbol{c}^T\boldsymbol{d} + \frac{1}{2}\boldsymbol{d}^T\boldsymbol{d}$$
$$\text{Subject to, } \boldsymbol{A}^T\boldsymbol{d} \leq \boldsymbol{b},\ \boldsymbol{N}^T\boldsymbol{d} = \boldsymbol{e} \tag{7.46}$$

Download free eBooks at bookboon.com

Since the QP subproblem represents a convex programming problem, a unique global minimum, if one exists, can be obtained. We further make the following observations regarding the QP problem:

1. From a geometric perspective, $\bar{f}$ represents the equation of a hypersphere with its center $-\boldsymbol{c}$, at and the search direction $\boldsymbol{d}$ points to the center of the hypersphere.
2. When there are no active constraints, application of FONC: $\frac{\partial \bar{f}}{\partial \boldsymbol{d}} = \boldsymbol{c} + \boldsymbol{d} = 0$, results in the search direction: $\boldsymbol{d} = -\boldsymbol{c}$, which conforms to the steepest descent direction.
3. When constraints are present, the QP solution amounts to projecting the steepest-descent direction onto the constraint hyperplane; the resulting search direction is termed as constrained steepest-descent (CSD) direction.

The QP subproblem can be analytically solved via the Lagrangian function approach. To do that, we add a slack variable $\boldsymbol{s}$ to the inequality constraint, and construct a Lagrangian function given as:

$$\mathcal{L}(\boldsymbol{d}, \boldsymbol{u}, \boldsymbol{v}) = \boldsymbol{c}^T\boldsymbol{d} + \tfrac{1}{2}\boldsymbol{d}^T\boldsymbol{d} + \boldsymbol{u}^T(\boldsymbol{A}^T\boldsymbol{d} - \boldsymbol{b} + \boldsymbol{s}) + \boldsymbol{v}^T(\boldsymbol{N}^T\boldsymbol{d} - \boldsymbol{e}) \tag{7.47}$$

Then, the KKT conditions for a minimum are:

$$\boldsymbol{\nabla}\mathcal{L} = \boldsymbol{c} + \boldsymbol{d} + \boldsymbol{A}\boldsymbol{u} + \boldsymbol{N}\boldsymbol{v} = \boldsymbol{0}, \qquad \boldsymbol{A}^T\boldsymbol{d} + \boldsymbol{s} = \boldsymbol{b}, \qquad \boldsymbol{N}^T\boldsymbol{d} = \boldsymbol{e}\,, \qquad \boldsymbol{u}^T\boldsymbol{s} = \boldsymbol{0}, \boldsymbol{u} \geq \boldsymbol{0}, \boldsymbol{s} \geq \boldsymbol{0} \tag{7.48}$$

Further, by writing $\boldsymbol{v} = \boldsymbol{y} - \boldsymbol{z}, \boldsymbol{y} \geq \boldsymbol{0}, \boldsymbol{z} \geq \boldsymbol{0}$, these conditions are expressed in matrix form as:

$$\begin{bmatrix} \boldsymbol{I} & \boldsymbol{A} & \boldsymbol{0} & \boldsymbol{N} & -\boldsymbol{N} \\ \boldsymbol{A}^T & \boldsymbol{0} & \boldsymbol{I} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{N}^T & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{d} \\ \boldsymbol{u} \\ \boldsymbol{s} \\ \boldsymbol{y} \\ \boldsymbol{z} \end{bmatrix} = \begin{bmatrix} -\boldsymbol{c} \\ \boldsymbol{b} \\ \boldsymbol{e} \end{bmatrix}, \text{ or } \boldsymbol{P}\boldsymbol{X} = \boldsymbol{Q} \tag{7.49}$$

where the complementary slackness conditions, $\boldsymbol{u}^T\boldsymbol{s} = \boldsymbol{0}$, translate as: $\boldsymbol{X}_i\boldsymbol{X}_{i+m} = 0, i = n+1, \cdots, n+m$. We note that solution to the above problem can be obtained via LCP framework (Sec. 5.7.1).

Once a search direction $\boldsymbol{d}$ has been determined, a step-size along $\boldsymbol{d}$ needs to be computed by solving the line search problem. We next discuss the descent function approach that is used to resolve the line search step in the SQP solution process.

7.6.1 Descent Function Approach

In SQP methods, the line search solution is based on minimization of a descent function that penalizes constraint violations. The following descent function has been proposed in literature (Arora, p. 521):

$$\Phi(\boldsymbol{x}) = f(\boldsymbol{x}) + RV(\boldsymbol{x}) \tag{7.50}$$

Download free eBooks at bookboon.com

where $f(\boldsymbol{x})$ represents the cost function value, $V(\boldsymbol{x})$ represents the maximum constraint violation, and $R > 0$ is a penalty parameter.

The descent function value at the current iteration is expressed as:

$$\Phi_k = f_k + RV_k, R = \max\{R_k, r_k\} \tag{7.51}$$

where R_k is the current value of the penalty parameter, r_k is the current sum of the Lagrange multipliers, and V_k is the maximum constraint violation in the current step. The latter parameters are computed as:

$$\begin{aligned} r_k &= \sum_{i=1}^{m} u_i^k + \sum_{j=1}^{p} \left|v_j^k\right| \\ V_k &= \max\{0;\ g_i,\ i = 1, \ldots, m;\ \left|h_j\right|,\ j = 1, \ldots, p\} \end{aligned} \tag{7.52}$$

where absolute values of the Lagrange multipliers and constraint violations for equality constraints are used. Next, the line search subproblem is defined as:

$$\min_{\alpha} \Phi(\alpha) = \Phi\left(\boldsymbol{x}^k + \alpha \boldsymbol{d}^k\right) \tag{7.53}$$

The above problem may be solved via the line search methods described in Sec. 7.2.

An algorithm for solving the SQP problem is presented below:

SQP Algorithm (Arora, p. 526)**:**

Initialize: choose $\boldsymbol{x}^0, R_0 = 1, \varepsilon_1 > 0, \varepsilon_2 > 0$.
For $k = 0,1,2, \ldots$

1. Compute $f^k,\ g_i^k, h_j^k, \boldsymbol{c}, b_i, e_j$; compute V_k.
1. Formulate and solve the QP subproblem to obtain $\boldsymbol{d}^k$ and the Lagrange multipliers $\boldsymbol{u}^k$ and $\boldsymbol{v}^k$.
2. If $V_k \leq \varepsilon_1$ and $\left\|\boldsymbol{d}^k\right\| \leq \varepsilon_2$, stop.
3. Compute R; formulate and solve line search subproblem to obtain α
4. Set $\boldsymbol{x}^{k+1} \leftarrow \boldsymbol{x}^k + \alpha \boldsymbol{d}^k,\ R_{k+1} \leftarrow R,\ k \leftarrow k + 1$.

It can be shown that the above algorithm is convergent, i.e., $\Phi\left(\boldsymbol{x}^k\right) \leq \Phi(\boldsymbol{x}^0)$, and that $\boldsymbol{x}^k$ converges to the KKT point in the case of general constrained optimization problems (Arora, p. 525).

Download free eBooks at bookboon.com

7.6.2 SQP with Approximate Line Search

The above SQP algorithm can be used with approximate line search methods, similar to Arjimo's rule (Sec. 7.2.2) as follows: let $t_j,\ j = 0,1,\ldots$ denote a trial step size, $\boldsymbol{x}^{k+1,j}$ denote the trial design point, $f^{k+1,j} = f(\boldsymbol{x}^{k+1,j})$ denote the function value at the trial solution, and $\Phi_{k+1,j} = f^{k+1,j} + RV_{k+1,j}$ denote the penalty function at the trial solution. The trial solution is required to satisfy the following descent condition:

$$\Phi_{k+1,j} + t_j \gamma \left\| \boldsymbol{d}^k \right\|^2 \leq \Phi_{k,j}, \qquad 0 < \gamma < 1 \tag{7.54}$$

where a common choice for γ is: $\gamma = \frac{1}{2}$. Further, $t_j = \mu^j,\ \mu = \frac{1}{2},\ j = 0,1,2,\ldots.$ The above descent condition ensures that the constraint violation decreases at each step of the method. The following example illustrates the application of approximate line search algorithm.

Example 7.7: Sequential Quadratic Programming with Approximate Line Search

We consider the above NL problem, given as:

$$\min_{x_1,x_2} f(x_1,x_2) = x_1^2 - x_1x_2 + x_2^2$$
subject to $g_1: 1 - x_1^2 - x_2^2 \leq 0,\ g_2: -x_1 \leq 0,\ g_3: -x_2 \leq 0.$

Download free eBooks at bookboon.com

where the gradient functions are computed as: $\nabla f^T = [2x_1 - x_2, 2x_2 - x_1]$, $\nabla g_1^T = [-2x_1, -2x_2]$, $\nabla g_2^T = [-1,0]$, $\nabla g_3^T = [0, -1]$.

Let $x^0 = (1,1)$; then, $f^0 = 1$, $\boldsymbol{c} = [1,\ 1]^T$, $g_1(1,1) = g_2(1,1) = g_3(1,1) = -1$. Since, at this point, there are no active constraints, $V_0 = 0$, the preferred search direction is: $\boldsymbol{d} = -\boldsymbol{c} = [-1,-1]^T$; the line search problem is defined as: $\min_{\alpha} \Phi(\alpha) = f(\boldsymbol{x}^0 + \alpha \boldsymbol{d}^0) = (1-\alpha)^2$. This problem can be analytically solved by setting $\Phi'(\alpha) = 0$, with the solution: $\alpha = 1$, resulting in $x^1 = (0,0)$; however, this analytical solution results in a large constraint violation that is undesired.

Use of the approximate line search method for the problem results in the following computations:

Let $t_0 = 1$, $R_0 = 10, \gamma = \mu = \frac{1}{2}$; then $\boldsymbol{x}^{1,0} = (0,0)$, $\|\boldsymbol{d}^0\|^2 = 2$, $f^{1,0} = 0$, $V_{1,0} = 1$, $\Phi_{1,0} = 10$, and the descent condition $\Phi_{1,0} + \frac{1}{2}\|\boldsymbol{d}^0\|^2 \leq \Phi_0 = 1$ is not met. We then try $t_1 = \frac{1}{2}$ to obtain: $\boldsymbol{x}^{1,1} = \left(\frac{1}{2}, \frac{1}{2}\right)$, $V_{1,1} = \frac{1}{2}$, $\Phi_{1,1} = 5\frac{1}{4}$, and the descent condition fails again; next, for $t_2 = \frac{1}{4}$, we get: $\boldsymbol{x}^{1,2} = \left(\frac{3}{4}, \frac{3}{4}\right)$, $V_{1,2} = 0$, $\Phi_{1,2} = \frac{9}{16}$, and the descent condition checks as: $\Phi_{1,2} + \frac{1}{8}\|\boldsymbol{d}^0\|^2 \leq \Phi_0$. Therefore, we set: $\alpha = t_2 = \frac{1}{4}$, $\boldsymbol{x}^1 = \boldsymbol{x}^{1,2} = \left(\frac{3}{4}, \frac{3}{4}\right)$ with no constraint violation.

Next, we discuss some modifications to the SQP method that aid in solution to the QP subproblem.

7.6.3 The Active Set Strategy

The computational cost of solving the QP subproblem can be substantially reduced by only including the active constraints in the subproblem. Accordingly, if the current design point $\boldsymbol{x}^k \in \Omega$, where Ω denotes the feasible region, then, for some small $\varepsilon > 0$, the set $\mathcal{J}_k = \{i: g_i^k > -\varepsilon; i = 1, \dots, m\} \cup \{j: j = 1, \dots, p\}$ denotes the set of potentially active constraints. In the event $\boldsymbol{x}^k \notin \Omega$, let the current maximum constraint violation be given as: $V_k = \max\{0;\ g_i^k,\ i = 1, \dots, m;\ |h_j^k|,\ j = 1, \dots, p\}$; then, the active constraint set includes: $\mathcal{J}_k = \{i: g_i^k > V_k - \varepsilon; i = 1, \dots, m\} \cup \{j: |h_j^k| > V_k - \varepsilon;\ j = 1, \dots, p\}$.

We may note that an inequality constraint at the current design point can be characterized in the following ways: as active (if $g_i^k = 0$), as ε-active (if $g_i^k > -\varepsilon$), as violated ($if\ g_i^k > 0$), or as inactive (if $g_i^k \leq -\varepsilon$); whereas, an equality constraint is either active ($h_j^k = 0$) or violated ($h_j^k \neq 0$).

The gradients of constraints not in $\mathcal{J}_k$ do not need to be computed, however, the numerical algorithm using the potential constraint strategy must be proved to be convergent. Further, from a practical point of view, it is desirable to normalize all constraints with respect to their limit values, so that a uniform ε value can be used to check for a constraint condition at the design point.

Download free eBooks at bookboon.com

Using the active set strategy, the active inequality constraints being known, they can be treated as equality constraints. We, therefore, assume that only equality constraints are present in the active set, and define the QP subproblem as:

$$\begin{aligned} &\min_{\boldsymbol{d}} \bar{f} = \boldsymbol{c}^T\boldsymbol{d} + \frac{1}{2}\boldsymbol{d}^T\boldsymbol{d} \\ &\text{Subject to: } \bar{\boldsymbol{N}}^T\boldsymbol{d} = \bar{\boldsymbol{e}} \end{aligned} \tag{7.55}$$

Then, using the Lagrangian function approach, the optimality conditions are given as: $\bar{\boldsymbol{N}}\boldsymbol{v} + \boldsymbol{c} + \boldsymbol{d} = \boldsymbol{0}$, $\bar{\boldsymbol{N}}^T\boldsymbol{d} - \bar{\boldsymbol{e}} = \boldsymbol{0}$. They can be simultaneously solved to eliminate the Lagrange multipliers as follows: from the optimality conditions we solve for $\boldsymbol{d}$ as: $\boldsymbol{d} = -\boldsymbol{c} - \bar{\boldsymbol{N}}\boldsymbol{v}$, and substitute it in the constraint equation to get: $\bar{\boldsymbol{N}}^T\bar{\boldsymbol{N}}\boldsymbol{v} = -\bar{\boldsymbol{N}}^T(\boldsymbol{c} + \boldsymbol{d})$. Next, we substitute $\boldsymbol{v}$ back in the optimality condition to get:

$$\boldsymbol{d} = -[\boldsymbol{I} - \bar{\boldsymbol{N}}(\bar{\boldsymbol{N}}^T\bar{\boldsymbol{N}})^{-1}\bar{\boldsymbol{N}}^T]\boldsymbol{c} + \bar{\boldsymbol{N}}(\bar{\boldsymbol{N}}^T\bar{\boldsymbol{N}})^{-1}\boldsymbol{e} \tag{7.56}$$

or, more compactly as: $\boldsymbol{d} = \boldsymbol{d}_1 + \boldsymbol{d}_2$, where $\boldsymbol{d}_1$ in the above expression defines a matrix operator: $\boldsymbol{P} = \boldsymbol{I} - \bar{\boldsymbol{N}}(\bar{\boldsymbol{N}}^T\bar{\boldsymbol{N}})^{-1}\bar{\boldsymbol{N}}^T$, $\boldsymbol{PP} = \boldsymbol{P}$, that projects the gradient of the cost function onto the tangent hyperplane defined by: $\{\boldsymbol{d}: \bar{\boldsymbol{N}}^T\boldsymbol{d} = 0\}$, which can also be obtained as a solution to the following minimization problem: $\min_{\boldsymbol{d}} \|\boldsymbol{c} - \boldsymbol{d}\|^2$ subject to $\bar{\boldsymbol{N}}^T\boldsymbol{d} = \boldsymbol{0}$ (Belegundu and Chandrupatla, p. 243).

The second part of $\boldsymbol{d}$ defines a vector that points toward the feasible region. Further, these two components are orthogonal, i.e., $\boldsymbol{d}_1^T\boldsymbol{d}_2 = 0$. Thus, we may interpret $\boldsymbol{d}$ as a combination of a cost reduction step $\boldsymbol{d}_1$ and a constraint correction step $\boldsymbol{d}_2$. Further, if there are no constraint violations, i.e., $\bar{\boldsymbol{e}} = \boldsymbol{0}$, then $\boldsymbol{d}_2 = \boldsymbol{0}$, and $\boldsymbol{d}$ aligns with the projected steepest descent direction.

7.6.4 SQP Update via Newton's Update

We observe that, from a computational point of view, Newton's method can be used to solve the SQP subproblem. In order to derive the SQP update via Newton's method, we consider the following design optimization problem involving equality constraints (Arora, p. 554):

$$\begin{aligned} &\min_{\boldsymbol{x}} f(\boldsymbol{x}) \\ &\text{Subject to: } h_i(\boldsymbol{x}) = 0,\ i = 1, \ldots, l \end{aligned} \tag{7.57}$$

The Lagrangian function for the problem is constructed as:

$$\mathcal{L}(\boldsymbol{x}, \boldsymbol{v}) = f(\boldsymbol{x}) + \boldsymbol{v}^T\boldsymbol{h}(\boldsymbol{x}) \tag{7.58}$$

The KKT conditions for a minimum are given as:

$$\nabla\mathcal{L}(\boldsymbol{x}, \boldsymbol{v}) = \nabla f(\boldsymbol{x}) + \boldsymbol{N}\boldsymbol{v} = \boldsymbol{0}, \boldsymbol{h}(\boldsymbol{x}) = \boldsymbol{0} \tag{7.59}$$

Download free eBooks at bookboon.com

where $\boldsymbol{N} = \nabla \boldsymbol{h}^T(\boldsymbol{x})$ is the Jacobian matrix whose ith columns represents the gradient ∇h_i. Next, Newton's method is employed to compute the change in the design variables and Lagrange multipliers as follows: using first order Taylor series expansion for $\nabla \mathcal{L}^{k+1}$ and $\boldsymbol{h}^{k+1}$, we obtain:

$$\begin{bmatrix} \nabla^2 \mathcal{L} & \boldsymbol{N} \\ \boldsymbol{N}^T & 0 \end{bmatrix}^k \begin{bmatrix} \Delta \boldsymbol{x} \\ \Delta \boldsymbol{v} \end{bmatrix}^k = -\begin{bmatrix} \nabla \mathcal{L} \\ \boldsymbol{h} \end{bmatrix}^k \tag{7.60}$$

The first equation above may be expanded as: $\nabla^2 \mathcal{L} \Delta \boldsymbol{x}^k + \boldsymbol{N}\left(\boldsymbol{v}^{k+1} - \boldsymbol{v}^k\right) = -\left(\nabla f^k(\boldsymbol{x}) + \boldsymbol{N}\boldsymbol{v}^k\right)$, and simplified as: $\nabla^2 \mathcal{L} \Delta \boldsymbol{x}^k + \boldsymbol{N}\boldsymbol{v}^{k+1} = -\nabla f^k(\boldsymbol{x})$, resulting in the following Newton-Raphson iteration:

$$\begin{bmatrix} \nabla^2 \mathcal{L} & \boldsymbol{N} \\ \boldsymbol{N}^T & 0 \end{bmatrix}^k \begin{bmatrix} \Delta \boldsymbol{x}^k \\ \Delta \boldsymbol{v}^{k+1} \end{bmatrix} = -\begin{bmatrix} \nabla f \\ \boldsymbol{h} \end{bmatrix}^k \tag{7.61}$$

It is interesting to note that the above result can also be obtained via a QP problem defined in terms of incremental variables where the QP problem is defined as follows:

$$\begin{aligned} &\min_{\Delta \boldsymbol{x}} \ \frac{1}{2} \Delta \boldsymbol{x}^T \ \nabla^2 \mathcal{L} \ \Delta \boldsymbol{x} + \nabla f^T \Delta \boldsymbol{x} \\ &\text{Subject to: } h_i(\boldsymbol{x}) + n_i^{\mathrm{T}} \Delta \boldsymbol{x} = 0, \ i = 1, \ldots, l \end{aligned} \tag{7.62}$$

The Lagrangian function for the problem is formulated as:

$$\mathcal{L}(\Delta \boldsymbol{x}, \boldsymbol{v}) = \frac{1}{2} \Delta \boldsymbol{x}^T \ \nabla^2 \mathcal{L} \ \Delta \boldsymbol{x} + \nabla f^T \Delta \boldsymbol{x} + \boldsymbol{v}^T (\boldsymbol{h} + \boldsymbol{N} \Delta \boldsymbol{x}) \tag{7.63}$$

Download free eBooks at bookboon.com

The resulting KKT conditions for an optimum are given as: $\nabla f + \nabla^2 \mathcal{L}\, \Delta \boldsymbol{x} + \boldsymbol{N}\boldsymbol{v} = \boldsymbol{0}, \boldsymbol{h} + \boldsymbol{N}\Delta \boldsymbol{x} = \boldsymbol{0}$. In matrix form, these KKT conditions are similar to those used in the Newton-Raphson update.

7.6.5 SQP with Hessian Update

The above Newton's implementation of SQP algorithm uses Hessian of the Lagrangian function for the update. Since Hessian computation is relatively costly, an approximate to the Hessian may instead be used. Towards that end, let $\boldsymbol{H} = \nabla^2 \mathcal{L}$, then the modified QP subproblem is defined as (Arora, p. 557):

$$\begin{aligned} &\min_{\boldsymbol{d}} \bar{f} = \boldsymbol{c}^T \boldsymbol{d} + \frac{1}{2} \boldsymbol{d}^T \boldsymbol{H} \boldsymbol{d} \\ &\text{Subject to, } \boldsymbol{A}^T \boldsymbol{d} \le \boldsymbol{b},\ \boldsymbol{N}^T \boldsymbol{d} = \boldsymbol{e} \end{aligned} \tag{7.64}$$

We note that quasi-Newton methods (Sec. 7.3.4) solve the unconstrained minimization problem by solving a set of linear equations given as: $\boldsymbol{H}^k \boldsymbol{d}^k = -\boldsymbol{c}^k$ for $\boldsymbol{d}^k$, where $\boldsymbol{H}^k$ where represents an approximation to the Hessian matrix. In particular, the popular BFGS method uses the following Hessian update:

$$\boldsymbol{H}^{k+1} = \boldsymbol{H}^k + \boldsymbol{D}^k + \boldsymbol{E}^k \tag{7.65}$$

where $\boldsymbol{D}^k = \frac{\boldsymbol{y}^k \boldsymbol{y}^{k^T}}{\boldsymbol{y}^{k^T} \boldsymbol{s}^k}$, $\boldsymbol{E}^k = \frac{\boldsymbol{c}^k \boldsymbol{c}^{k^T}}{\boldsymbol{c}^{k^T} \boldsymbol{d}^k}$, $\boldsymbol{s}^k = \alpha_k \boldsymbol{d}^k$, $\boldsymbol{y}^k = \boldsymbol{c}^{k+1} - \boldsymbol{c}^k$, $\boldsymbol{c}^k = \nabla f(\boldsymbol{x}^k)$.

Next, the BFGS Hessian update is modified to apply to the constrained optimization problems as follows: let $\boldsymbol{s}^k = \alpha_k \boldsymbol{d}^k$, $\boldsymbol{z}^k = \boldsymbol{H}^k \boldsymbol{s}^k$, $\boldsymbol{y}^k = \nabla \mathcal{L}(\boldsymbol{x}^{k+1}) - \nabla \mathcal{L}(\boldsymbol{x}^k)$, $\boldsymbol{s}^{k^T} \boldsymbol{y}^k = \xi_1$, $\boldsymbol{s}^{k^T} \boldsymbol{z}^k = \xi_2$; further, define: $\boldsymbol{w}^k = \theta \boldsymbol{y}^k + (1-\theta)\boldsymbol{z}^k$, where $\theta = \min\left\{1, \frac{0.8\xi_2}{\xi_2 - \xi_1}\right\}$, $\boldsymbol{s}^{k^T} \boldsymbol{w}^k = \xi_3$; then, the Hessian update is given as: $\boldsymbol{H}^{k+1} = \boldsymbol{H}^k + \boldsymbol{D}^k - \boldsymbol{E}^k$, $\boldsymbol{D}^k = \frac{1}{\xi_3} \boldsymbol{y}^k \boldsymbol{y}^{k^T}$, $\boldsymbol{E}^k = \frac{1}{\xi_2} \boldsymbol{z}^k \boldsymbol{z}^{k^T}$.

The modified SQP algorithm is given as follows:

Modified SQP Algorithm (Arora, p. 558)**:**

Initialize: choose $\boldsymbol{x}^0, R_0 = 1$, $\boldsymbol{H}^0 = I$; $\varepsilon_1, \varepsilon_2 > 0$.
For $k = 0,1,2, \ldots$

1. Compute f^k, $g_i^k, h_j^k, \boldsymbol{c}, b_i, e_j$, and V_k. If $k > 0$, compute $\boldsymbol{H}^k$
2. Formulate and solve the modified QP subproblem for search direction $\boldsymbol{d}^k$ and the Lagrange multipliers $\boldsymbol{u}^k$ and $\boldsymbol{v}^k$.
3. If $V_k \le \varepsilon_1$ and $\|\boldsymbol{d}^k\| \le \varepsilon_2$, stop.
4. Compute R; formulate and solve line search subproblem to obtain α
5. Set $\boldsymbol{x}^{k+1} \leftarrow \boldsymbol{x}^k + \alpha \boldsymbol{d}^k$, $R_{k+1} \leftarrow R$, $k \leftarrow k+1$.

Download free eBooks at bookboon.com

An example for SQP with Hessian update is presented below.

Example 7.8: SQP with Hessian Update

As an example, we consider the above NL problem, given as:

$\min_{x_1,x_2} f(x_1,x_2) = x_1^2 - x_1x_2 + x_2^2,$
subject to $g_1: 1 - x_1^2 - x_2^2 \le 0;\ g_2: -x_1 \le 0,\ g_3: -x_2 \le 0,$

The objective and constraint gradients for the problem are obtained as:

$$\nabla f^T = [2x_1 - x_2, 2x_2 - x_1],\ \nabla g_1^T = [-2x_1, -2x_2],\ \nabla g_2^T = [-1,0],\ \nabla g_3^T = [0,-1].$$

To proceed, let $x^0 = (1,1)$, so that, $f^0 = 1,\ g_1(1,1) = g_2(1,1) = g_3(1,1) = -1$; since all constraints are initially inactive, the preferred search direction is: $\boldsymbol{d} = -\boldsymbol{c} = [-1,-1]^T$; then, using approximate line search we obtain: $\alpha = \frac{1}{4}$, leading to: $\boldsymbol{x}^1 = \left(\frac{3}{4}, \frac{3}{4}\right)$.

For the Hessian update, we have: $f^1 = 0.5625,\ g_1 = -0.125,\ g_2 = g_3 = -0.75;\ \boldsymbol{c}^1 = [0.75, 0.75]$; and, for $\alpha = 0.25,\ \boldsymbol{s}^0 = [-0.25,-0.25] = \boldsymbol{z}^0 = \boldsymbol{y}^0,\ \xi_1 = \xi_2 = 0.125,\ \theta = 1,\ \boldsymbol{w}^0 = \boldsymbol{y}^0,\ \xi_3 = \xi_1$; therefore, Hessian update is computed as: $\boldsymbol{D}^0 = 8\begin{bmatrix}1 & 1\\ 1 & 1\end{bmatrix},\ \boldsymbol{E}^0 = 8\begin{bmatrix}1 & 1\\ 1 & 1\end{bmatrix},\ \boldsymbol{H}^1 = \boldsymbol{H}^0.$

For the next step, the QP problem is defined as:

$\min_{d_1,d_2} \bar{f} = \frac{3}{4}(d_1 + d_2) + \frac{1}{2}(d_1^2 + d_2^2)$
Subject to: $-\frac{3}{2}(d_1 + d_2) \le 0, -d_1 \le 0, -d_2 \le 0$

Using a Lagrangian function approach, the solution is found from application of KKT conditions, which results in the following systems of equations: $\boldsymbol{P}\boldsymbol{x} = \boldsymbol{q}$, where $\boldsymbol{x}^T = [d_1, d_2, u_1, u_2, u_3, s_1, s_2, s_3]$, and,

$$\boldsymbol{P} = \begin{bmatrix} 1 & 0 & -1.5 & -1 & 0 & 0 & 0 & 0\\ 0 & 1 & -1.5 & 0 & -1 & 0 & 0 & 0\\ -1.5 & -1.5 & 0 & 0 & 0 & 1 & 0 & 0\\ -1 & 0 & 0 & 0 & 0 & 0 & 1 & 0\\ 0 & -1 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix},\ \boldsymbol{q} = \begin{bmatrix} -0.75\\ -0.75\\ 0.125\\ 0.75\\ 0.75 \end{bmatrix}$$

The complementary slackness conditions are given as: $u_i s_i = 0,\ i = 1,2,3$. The solution found from the simplex method is given as: $\boldsymbol{x}^T = [0.188, 0.188, 0,\ 0,\ 0, 0.125, 0.75, 0.75]$. We note that in this case as the number of variables is small, taking the complementarity conditions into account, there are eight basic solutions, only one of which is feasible and is given as: $\boldsymbol{X}^T = [0.188, 0.188, 0,\ 0,\ 0, 0.125, 0.75, 0.75]$.

Download free eBooks at bookboon.com

8 References

Arora, JS 2004, *Introduction to Optimum Design,* 2nd edn, Elsevier Academic Press, San Diego, CA.

Belegundu, AD and Chandrupatla TR 2012, *Optimization Concepts and Applications in Engineering*, 2nd edn (reprinted), Cambridge University Press, New York.

Boyd, S & Vandenberghe, L 2004, *Convex Optimization*, Cambridge University Press, New York.

Chong, EKP & Zak, SH 2013, *An Introduction to Optimization*, 4th edn. John Wiley & Sons, New Jersey.

Eisenbrand, F, Course notes for linear and discrete optimization, https://class.coursera.org/linearopt-001/lecture/index

Ferris, MC, Mangasarian, OL & Wright, SJ 2007, Linear Programming with Matlab, SIAM, Philadelphia, PA

Ganguli, R 2012, *Engineering Optimiztion A Modern Approach*, Universities Press, Hyderabad (India).

Griva, I, Nash, SG & Sofer, A 2009, *Linear and Nonlinear Optimization*, 2nd edn, SIAM, Philadelphia, PA.

Hager, WW & Zhang, H-C 2006, 'A survey of nonlinear conjugate gradient methods', *Pacific Journal of Optimization*, vol. 2, pp. 35–58.

Hemmecke, R, Lecture notes on discrete optimization, https://www-m9.ma.tum.de/foswiki/pub/SS2011/DiscOpt/DiscOpt.pdf

Kelly, CT 1995, *Iterative Methods for Linear and Nonlinear Equations*, SIAM, Philadelphia, PA.

Luenberger, DG &Ye, Y 2008, *Linear and Nonlinear Programming*, 3rd edn, Springer, New York.

Pedregal, P 2004, *Introduction to Optimization*, Springer-Verlag, New York.

Sierksma, G 2002, *Linear and Integer Programming: Theory and Practice*, 2nd edn, Marcel Dekker, Monticello, NY.

Vanderbei, RJ 2007, *Linear Programming: Foundations and Extensions*, 3rd edn, Springer, New York.

Yang, X-S 2010, *Engineering Optimization*, John Wiley & Sons, New Jersey.

Download free eBooks at bookboon.com