Deterministic Global Optimization

Nonconvex Optimization and Its Applications

Volume 37

Managing Editors:

Panos Pardalos
University of Florida, U.S.A.

Reiner Horst
University of Trier, Germany

Advisory Board:

J. R. Birge
University of Michigan, U.S.A.

Ding-Zhu Du
University of Minnesota, U.S.A.

C. A. Floudas
Princeton University, U.S.A.

J. Mockus
Stanford University, U.S.A.

H. D. Sherali
Virginia Polytechnic Institute and State University, U.S.A.

G. Stavroulakis
Technical University Braunschweig, Germany

The titles published in this series are listed at the end of this volume.

Deterministic Global Optimization

Theory, Methods and Applications

by

Christodoulos A. Floudas

Department of Chemical Engineering,
Princeton University,
Princeton, New Jersey, U.S.A.

SPRINGER-SCIENCE+BUSINESS MEDIA, B.V.

Library of Congress Cataloging-in-Publication Data

Floudas, Christodoulos A.
Deterministic global optimization : theory, methods, and applications / by Christodoulos A. Floudas.
p. cm. -- (Nonconvex optimization and its applications ; v. 37)
Includes bibliographical references and index.
ISBN 978-1-4419-4820-5 ISBN 978-1-4757-4949-6 (eBook)
DOI 10.1007/978-1-4757-4949-6
1. Mathematical optimization. 2. Nonlinear programming. I. Title. II. Series.

QA402.5 .F586 1999
519.3--dc21

99-047045

ISBN 978-1-4419-4820-5

Printed on acid-free paper

All Rights Reserved
© 2000 Springer Science+Business Media Dordrecht
Originally published by Kluwer Academic Publishers in 2000
No part of the material protected by this copyright notice may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying, recording or by any information storage and retrieval system, without written permission from the copyright owner.

This book is dedicated to the
memory of my beloved father
Achilleus Christodoulos Floudas

Contents

Preface xiii

1. INTRODUCTION 1
   1.1 Motivation 1
   1.2 Structure of Global Optimization Models 3
   1.3 Illustrative Applications of Global Optimization 4
   1.4 Deterministic Global Optimization : Approaches and Classes of Problems 18
   1.5 Complexity Analysis 19
   1.6 Research Areas and Important Applications 23
   1.7 Publications and Conferences 25
   1.8 Chemical Engineering Contributions 27
   1.9 Scope of the Book 32

2. BASIC CONCEPTS OF GLOBAL OPTIMIZATION 33
   2.1 Convex Sets 33
   2.2 Convex and Concave Functions 35
   2.3 Quasi-Convex and Pseudo-Convex Functions 42
   2.4 Convex Envelopes 45
   2.5 Local, Global, ϵ-Global Minima 57
   2.6 Difference of Convex (D.C.) Functions 58

Part I Biconvex and Bilinear Problems

3. THE GOP PRIMAL - RELAXED DUAL DECOMPOSITION APPROACH : THEORY 67
   3.1 Problem Statement 68
   3.2 Duality Theory 70
   3.3 Mathematical Properties 73
   3.4 Infeasible Primal Problems 82
   3.5 The Global OPtimization (GOP) Algorithm 84
   3.6 Finite ϵ-Convergence and ϵ-Global Optimality 87
   3.7 Illustrative Example 90

3.8 The GOP for Quadratic Problems with Linear Constraints 95
3.9 The GOP for Quadratically Constrained Problems 96
3.10 The GOP for Univariate Polynomial Problems 98
3.11 Additional Properties of the Lagrange Function 105
3.12 The GOP in a Branch and Bound Framework 114
3.13 Reformulation Of The Relaxed Dual As A Single MILP Problem 126
3.14 A Linear Branching Scheme for the GOP Algorithm 134

4. THE GOP APPROACH : IMPLEMENTATION AND COMPUTATIONAL STUDIES 141
4.1 Implementation of the GOP and GOP/MILP Algorithms 141
4.2 Computational Studies 152
4.3 Summary 168

5. THE GOP APPROACH IN BILEVEL LINEAR AND QUADRATIC PROBLEMS 173
5.1 Introduction 173
5.2 Bilevel linear programming problem 175
5.3 Modified GOP Algorithm 181
5.4 bilevel linear-quadratic and bilevel quadratic-quadratic problems 187

6. THE GOP APPROACH IN PHASE AND CHEMICAL EQUILIBRIUM PROBLEMS 193
6.1 Introduction and Formulation of Minimizing the Gibbs Free Energy 193
6.2 Application of the GOP to NRTL Equation 198
6.3 Computational Studies 212
6.4 Tangent Plane Stability Criterion 225
6.5 Application to the NRTL Equation 229
6.6 Illustrative Example for the NRTL : n-Butyl-Acetate – Water 234

7. THE GOP APPROACH : DISTRIBUTED IMPLEMENTATION 243
7.1 Critical Components of the Distributed GOP Approach 243
7.2 Large Scale Indefinite Quadratic Problems 247
7.3 Large Scale Pooling and Blending Problems 251

Part II Signomial Problems

8. GENERALIZED GEOMETRIC PROGRAMMING : THEORY 257
8.1 Introduction 257
8.2 Theoretical Analysis 261
8.3 Global Optimization Algorithm 276
8.4 Proof of Convergence to the Global Minimum 282
8.5 Illustrative Example 283

9. GENERALIZED GEOMETRIC PROGRAMMING : COMPUTATIONAL STUDIES 289
9.1 Chemical Engineering Design Examples 289
9.2 Robust Stability Analysis 297

Part III Towards General Twice Differentiable NLPs

10. FROM BICONVEX TO GENERAL TWICE DIFFERENTIABLE NLPS 309
10.1 Theoretical Results : GOP to αBB 309
10.2 Estimation of the Parameter α 311

11. THE αBB FOR BOX CONSTRAINED TWICE-DIFFERENTIABLE NLPS : THEORY 315
11.1 Formulation and D.C. Transformation 315
11.2 Convex Lower Bounding Function 318
11.3 Global Optimization Algorithm αBB 323
11.4 Steps of the Global Optimization Algorithm αBB 324
11.5 Proof of Convergence to the Global Minimum 326
11.6 Complexity Analysis 329

12. THE αBB FOR CONSTRAINED TWICE -DIFFERENTIABLE NLPS : THEORY 333
12.1 Formulation and Principles of the αBB Global Optimization Algorithm 334
12.2 Underestimation Strategies 334
12.3 Overall Valid Convex Underestimator 340
12.4 Rigorous calculation of α for general NLPs 342
12.5 Illustrative Example 355
12.6 Global Optimization Algorithm, αBB 361
12.7 Geometrical Interpretation of the αBB Approach 365

13. COMPUTATIONAL STUDIES OF THE αBB APPROACH 377
13.1 Algorithmic Issues and Development of the αBB 377
13.2 Implementation of the αBB 382
13.3 Computational Studies of the αBB Approach 384

14. GLOBAL OPTIMIZATION IN MICROCLUSTERS 403
14.1 Introduction 403
14.2 Problem Definition 407
14.3 Novel D.C. Transformation via Eigenvalue Analysis 409
14.4 Mathematical Model 418
14.5 Algorithmic Procedure 423
14.6 Small Cluster Examples 425
14.7 A Relaxation of the Global Optimization Approach 426
14.8 Discussion on Magic Numbers and Interatomic Spacing 431
14.9 Summary 432

15. THE αBB APPROACH IN MOLECULAR STRUCTURE PREDICTION 435
15.1 Introduction and Background 435
15.2 Problem Definition 438
15.3 Problem Reformulation in Internal Coordinates 439
15.4 Convex Lower Bounding Function L of V 444
15.5 Computational Study : Propanal 447

16. THE αBB APPROACH IN PROTEIN FOLDING 451
16.1 Introduction 451
16.2 Mathematical Modeling 452
16.3 Global Optimization 459
16.4 Computational Studies : Unsolvated Oligopeptides 466
16.5 Computational Studies : Solvated Oligopeptides 471

17. THE αBB APPROACH IN PEPTIDE DOCKING 481
17.1 Introduction 481
17.2 Peptide Docking Methods 483
17.3 Problem Definition 485
17.4 Outline of Approach 488
17.5 Mathematical Modeling and Problem Formulation 490
17.6 Global Optimization Approach and Its Interface 493
17.7 Computational Studies and Discussion 498

18. THE αBB APPROACH IN BATCH DESIGN UNDER UNCERTAINTY 507
18.1 Introduction 507
18.2 Multiproduct Design Problem 509
18.3 Lower Bounding Problem Formulations 517
18.4 Global Optimization Algorithm - Modified αBB 524
18.5 Illustrative Example 528
18.6 Mixed-Product Campaign Formulation 530
18.7 Computational Studies 533
18.8 Comparison to Alternative Underestimating Approaches 538

19. THE αBB APPROACH IN PARAMETER ESTIMATION 543
19.1 Introduction 543
19.2 Maximum Likelihood Estimation 545
19.3 Global Optimization Framework 548
19.4 Global Optimization Algorithm 551
19.5 Computational Studies 554

Part IV Nonlinear and Mixed-Integer Optimization

20. INTRODUCTION TO NONLINEAR AND MIXED-INTEGER OPTIMIZATION 571
20.1 Motivation 571
20.2 Mathematical Description 575
20.3 Overview of Local Optimization Approaches for Convex MINLP Models 576
20.4 Overview of Global Optimization Approaches for Nonconvex MINLP Models 579

21. THE SMIN-αBB APPROACH : THEORY AND COMPUTATIONS 587
21.1 Mathematical Formulation 587
21.2 Generation of Valid Lower Bounds 588

21.3 Generation of Upper Bounds 590
21.4 Selection of Branching Variables 591
21.5 Updates of Variable Bounds 592
21.6 The SMIN-αBB Algorithm 595
21.7 Computational studies 598

22. THE GMIN-αBB APPROACH : THEORY AND COMPUTATIONS 617
22.1 Mathematical Formulation 618
22.2 Generation of Lower Bounds 618
22.3 Generation of Upper Bounds 621
22.4 Selection of Branching Variables 622
22.5 Updates of Variable Bounds 622
22.6 The GMIN-αBB Algorithm 624
22.7 Computational studies 624

Part V Nonlinear Constrained Systems of Equations

23. ALL SOLUTIONS OF NONLINEAR CONSTRAINED SYSTEMS OF EQUATIONS 641
23.1 Introduction 641
23.2 Problem Definition 643
23.3 Convex Lower Bounding 646
23.4 procedure for enclosing all solutions 653
23.5 Computational Studies 657

24. LOCATING ALL HOMOGENEOUS AZEOTROPES 667
24.1 Introduction 667
24.2 Problem Description 668
24.3 Mathematical Formulation 670
24.4 Activity Coefficient Models 672
24.5 Computational Studies 692

References 699

Preface

The vast majority of important applications in science, engineering and applied science are characterized by the existence of multiple minima and maxima, as well as first, second and higher order saddle points. The area of *Deterministic Global Optimization* introduces theoretical, algorithmic and computational advances that (i) address the computation and characterization of global minima and maxima, (ii) determine valid lower and upper bounds on the global minima and maxima, and (iii) address the enclosure of all solutions of nonlinear constrained systems of equations. Global optimization applications are widespread in all disciplines and they range from atomistic or molecular level to process and product level representations.

The primary goal of this book is three fold : first, to introduce the reader to the basics of deterministic global optimization; second, to present important theoretical and algorithmic advances for several classes of mathematical problems that include biconvex and bilinear problems, signomial problems, general twice differentiable nonlinear problems, mixed integer nonlinear problems, and the enclosure of all solutions of nonlinear constrained systems of equations; and third, to tie the theory and methods together with a variety of important applications.

Chapter 1 motivates first the reader for studying global optimization, introduces the major classes of mathematical models studied in this book, and introduces the reader to several motivating examples for global optimization that arise in important application areas. Then, it provides an overview of the global optimization approaches, discusses the basics of complexity analysis, presents the research areas and applications, the textbook and monograph publications, the conferences, and outlines the contributions from the chemical engineering field.

Chapter 2 presents the basics in deterministic global optimization that include convex sets; convex, concave and nonconvex functions; quasiconvex and quasiconcave functions; pseudoconvex and pseudoconcave functions; convex envelopes; local and global minima; and difference of convex (d.c.) functions.

For the remaining chapters, the book contains the following five main parts:

- Part I : Biconvex and Bilinear Problems
- Part II : Signomial Problems
- Part III : General Twice Differentiable NLP Problems
- Part IV : Nonlinear and Mixed Integer Optimization MINLP Problems
- Part V : Nonlinear Constrained Systems of Equations

Part I, comprised of five chapters, focuses on the global optimization of biconvex and bilinear mathematical models. Chapter 3 introduces the theory of the GOP primal relaxed dual decomposition algorithm and its application to quadratic programming problems, quadratically constrained problems and polynomial problems. In addition, it presents the GOP approach through a branch and bound framework, discusses a reformulation of the relaxed dual as a single MILP and introduces a linear branching scheme. Chapter 4 describes the implementation of the GOP and presents a variety of computational studies. Chapter 5 extends the theory of the GOP to bilevel linear and quadratic optimization problems. Chapter 6 introduces the phase and chemical equilibrium problem and the application of the GOP to the minimization of the Gibbs free energy and the tangent plane stability criterion problems. Chapter 7 discusses the distributed implementation of the GOP and its application to large scale pooling problems and indefinite quadratic programming problems.

Part II, consisting of two chapters, addresses the global optimization of generalized geometric programming problems. Chapter 8 introduces the theoretical analysis for signomial problems that includes the exponential transformation, the convex lower bounding approach, the updates of the bounds, the monotonicity analysis, and the branch and bound framework, while Chapter 9 presents computational studies on chemical engineering design and robust stability analysis problems with real parametric uncertainty.

Part III, comprised of ten chapters, deals with general twice differentiable nonlinear optimization NLP problems. Chapter 10 introduces the key theoretical developments that set the basis for the transition from biconvex and bilinear problems to general twice differentiable NLPs. Chapter 11 presents the theory of the αBB global optimization approach for box constrained twice differentiable NLPs. Chapter 12 introduces the theory of the αBB global optimization approach for general constrained NLPs and presents several rigorous methods for the calculation of the parameters α. Chapter 13 discusses the implementation of the αBB and presents a variety of computational studies. Chapter 14 introduces the structure prediction of clusters and presents a global optimization approach based on the principles of the GOP and the αBB approaches. Chapter 15 focuses on the structure prediction of small acyclic molecules via the αBB global optimization approach. Chapter 16 introduces the protein folding problem and discusses a novel framework based on the αBB for the structure prediction of unsolvated and solvated peptides. Chapter 17 introduces the peptide docking problem and presents a decomposition and global optimization

approach for the interactions of pocket 1 of the HLA-DRB1 protein with small virus peptides. Chapter 18 focuses on the batch design under uncertainty of multiproduct and multipurpose plants and presents a global optmization framework based on the αBB and exact calculations of the α parameters. Chapter 19 introduces the parameter estimation problem of nonlinear algebraic models via the error in variables approach and presents a global optimization approach based on the αBB principles.

Part IV, consisting of three chapters, focuses on mixed integer nonlinear optimization MINLP problems. Chapter 20 presents an introduction to MINLPs and an overview of the local MINLP approaches and the global MINLP methods. Chapter 21 presents the theory and computational studies of the SMIN-αBB global optimization approach which is applicable to special structure MINLP problems, while Chapter 22 introduces the theory and applications of the GMIN-αBB approach which can be applied to general MINLPs.

Part V, comprised of two chapters, addresses the enclosure of all solutions of nonlinear constrained systems of equations. Chapter 23 presents the theory and algorithmic aspects of a deterministic global optimization approach along with several computational studies taken from the literature. Chapter 24 introduces the problem of locating all azeotropes and the application of the global optimization approach to the enclosure of all homogeneous azeotropes.

Special efforts have been made so as to make this book self-contained, introduce the reader gradually to the basics of deterministic global optimization, and subsequently present the new theoretical and algorithmic developments for the five classes of optimization problems discussed in Parts I-V. These theoretical advances are complemented by a gradual exposition to computational studies that range from literature problems to important applications that arise in a variety of disciplines including applied mathematics, chemical engineering, computational chemistry, computer science, molecular biology, operations research and economics.

This book grew out of more than ten years of research, graduate courses and seminars, short courses, and talks on the subject. It has been prepared keeping in mind that it can be used as a textbook for graduate level courses and as a desk reference. All chapters have several illustrations and geometrical interpretations of the theoretical results presented, while the presented applications demonstrate the potential of these global optimization developments. Furthermore, the theoretical and algorithmic advances have been implemented in the software packages cGOP for biconvex and bilinear optimization problems, αBB for general twice differentiable NLPs, MINOPT for convex mixed integer nonlinear optimization problems, SMIN-αBB and GMIN-αBB for nonconvex mixed integer nonlinear optimization problems, GLOPEQ and EQUISTAR for phase and chemical equilibrium, GLO-FOLD for protein folding, and GLO-DOCK for peptide docking. The interested reader may find additional information on the software packages at the homepage of my research group at http://titan.princeton.edu.

A number of individuals and institutions deserve acknowledgement. *First*, I thank my doctoral thesis advisor, Professor Ignacio E. Grossmann, Head of the Department of Chemical Engineering at Carnegie Mellon University for the graduate education and challenges in the topics of energy integration and uncertainty analysis during my 1982-85 Ph.D. study, as well as for the preparation and support in my academic endeavors. *Second*, I thank Professor Pablo G. Debenedetti, Chairman of the Department of Chemical Engineering at Princeton University for his support and encouragement for this scholarly effort. *Third*, I express my gratitude to Professor Iacovos Vasalos, Director of the Chemical Process Engineering Research Institute, CPERI, in Thessaloniki, Greece, for making my sabbatical stay at CPERI during the 1998 fall semester productive and memorable. *Fourth*, I am indebted to all my doctoral students and postdoctoral associates. In particular, special thanks go to Amy Ciric and Avanish Aggarwal who studied the global optimum search GOS approach; Vishy Visweswaran who investigated the GOP approach for biconvex and bilinear optimization problems and studied a variety of formulations, implementations and applications; Conor McDonald who introduced the first rigorous global optimization methods for phase and chemical equilibrium problems; Costas Maranas who investigated the αBB approach for box constrained NLPs, applied it to the structure prediction of clusters and molecules, extended the αBB to general constrained NLPs with Ioannis Androulakis, studied a global optimization approach for generalized geometric programming problems, and introduced a deterministic global optimization approach for the enclosure of all solutions of nonlinear systems of equations; Ioannis Androulakis who studied the distributed versions of the GOP and the αBB approach, the first approaches for oligopeptide folding with Costas Maranas, and global optimization for the peptide docking problem with Marianthi Ierapetritou and Heather Schafroth; Claire Adjiman who investigated several rigorous methods for the calculation of the α parameters, introduced the SMIN-αBB and GMIN-αBB approaches for nonconvex MINLPs and created the implementation of the αBB; Carl Schweiger who developed a modeling language and algorithmic framework, MINOPT, for the solution of convex MINLPs with differential and algebraic constraints; Steve Harding who studied rigorous global optimization approaches for the design under uncertainty of batch processes, the location of all homogeneous, heterogeneous and reactive azeotropes, and for the phase and chemical equilibrium problems that employ equations of state; Marianthi Ierapetritou who studied with Vishy Visweswaran a global optimization approach for bilevel linear and quadratic problems, and investigated with Zeynep Gümüş global optimization approaches for the feasibility test and flexibility index problems; John Klepeis who investigated novel approaches for the protein folding, free energy calculations, and the NMR structure prediction based on atomistic level modeling and the αBB approach; Randy Esposito who studied a global optimization approach for the parameter estimation, data reconcilliation and optimal control problems of nonlinear algebraic and differential-algebraic models based on the principles of the αBB approach; and Karl Westerberg

who investigated a deterministic global optimization method for the location of all transition states of small molecules and oligopeptides, and studied the reaction pathways and dynamics in protein folding. *Fifth*, I am indebted to Professors Ignacio Grossmann, Rex Reklaitis, Roger Sargent, Warren Seider, George Stephanopoulos, and Art Westerberg, who have provided inspiration in my quest for developing rigorous theories, methods, algorithms and implementations for important global optimization applications that were considered intractable a decade ago. *Sixth*, I have benefitted a lot from my collaborations with Dr. Jeffrey Abell, Professor Aharon Ben-Tal, Professor Jay Bailey, Dr. Stanley Cullick, Dr. Vassily Hatzimanikatis, Dr. Donald Jones, Professor John Lambris, Dr. Wenbin Liu, Professor William Luyben, Professor Dimitris Monos, Dr. Dimitris Morikis, Professor John Mulvey, Professor Arnold Neumaier, Professor Panos Pardalos, Professor Efstratios Pistikopoulos, Professor Herschel Rabitz, Professor Warren Seider, Dr. Sriram Vasantharajan, Professor Richard Yetter, and Professor Sanjo Zlobec. *Seventh*, special thanks and appreciation go to Professor Panos Pardalos and my doctoral students, Randy Esposito, Zeynep Gümüş, Steve Harding, John Klepeis, Xiaoxia Lin, and Clifford Meyer, for the excellent feedback, very careful reading of several drafts of this book, and assistance with the figures.

I am grateful to the funding sources that include the National Science Foundation, the Air Force Office of Scientific Research, the National Institutes of Health, as well as a number of industrial organizations that include Exxon Foundation, General Motors, Delphi Corporation, Mobil Technology Company, Mobil Foundation, Shell Development Company and Eastman Kodak. Special thanks go to the 1988 Presidential Young Investigator Award which allowed me, at the early stage of my academic career, to spend the resources and efforts in the considered as intractable research domain of global optimization, and to the 1996 Bodossaki Foundation Award in Applied Sciences for recognizing the contributions.

I am also thankful for the professional assistance of the staff at Kluwer Academic Publishers, and especially the senior editor John Martindale.

Finally, I am indebted to my wife Fotini and daughter Ismini for tolerating my long working hours even during my sabbatical leave and complementing their support and encouragement with entertaining humor, to my mother Ismini and my parents in law Petro and Ioanna Baba for their never ending genuine love and support, and to my parents in law and relatives in Thessaloniki, Greece for their hospitality during the 1998 fall semester sabbatical.

Christodoulos A. Floudas
Princeton, New Jersey
July 1999

1 INTRODUCTION

This chapter provides an introduction and overview of the activities in the area of *Deterministic Global Optimization.* Section 1.1 introduces the key issues, definitions, and motivation of the global optimization problems and contrasts them with the local optimization problems. Section 1.2 presents the structure of the three classes of mathematical models that we focus on this book. Section 1.3 presents a number of illustrative examples of important global optimization applications. Section 1.4 discusses the approaches that have been proposed in deterministic global optimization and the classes of problems that are addressed. Section 1.5 presents the basic definitions of complexity theory and discusses the worst case difficulties that one faces with global and local optimization problems based on complexity analysis results. Section 1.6 provides a large number of research areas that cross several disciplines and a number of important applications. Section 1.7 discusses the activities in the area of global optimization that consist of books published, conferences held, and the introduction of a journal and a book series. Section 1.8 presents an overview of the contributions that came from the field of chemical engineering. Finally, section 1.9 presents an outline of the five parts of this book.

1.1 MOTIVATION

Global optimization addresses the computation and characterization of global optima (i.e., minima and maxima) of nonconvex functions constrained in a

C. A. Floudas, *Deterministic Global Optimization*
© Springer Science+Business Media Dordrecht 2000

specified domain. Given an objective function f that is to be minimized and a set of equality and inequality constraints S, the area of *Deterministic Global Optimization* focuses on the following important issues :

(1) Determine a global minimum of the objective function f (i.e., f has the lowest possible value in S) subject to the set of constraints S;

(2) Determine *lower* and *upper bounds* on the global minimum of the objective function f on S that are valid for the whole feasible region S;

(3) Enclose all solutions of the set of equality and inequality constraints S.

In contrast to the aforementioned objectives of the area of deterministic global optimization, the area of *local optimization* aims at determining a feasible solution that is a local minimum of the objective function f on S (i.e., it is a minimum in its neighborhood but not necessarily of the lowest value for the objective function f). As a result, in general nonlinear optimization problems where there exist multiple local minima, a local minimum (and also any function evaluation point of f) represents only an upper bound on the global minimum of the objective function f on S.

It is remarkable that despite the fact that the vast majority of the challenging and important problems that arise in science, applied science, and engineering exhibit nonconvexities and hence multiple minima, there has been relatively little effort devoted to the area of global optimization in comparison to the developments in the area of local optimization. This is in part attributed to the use of local optimization techniques as components of global optimization approaches and in part to the difficulties that arise in the development of global optimization methods.

The world of applications though is no longer satisfied with the *good quality* local optimum solutions from the use of local optimization methods, but instead researchers and practitioners are interested in knowing the margins of improvement in the obtained solutions (i.e., valid upper and lower bounds) and if possible the determination of the best possible (i.e., global) optimum solution. Furthermore, there is a plethora of important research areas and problems such as the protein folding and peptide docking in computational chemistry and molecular biology, and phase and chemical equilibrium in thermodynamics where the physical interpretation of the global optimum solution can differ dramatically from the local optimum solutions. Therefore, there is a great demand and need for the development and effective application of rigorous global optimization methods.

The theoretical and algorithmic advances that have taken place in the area of deterministic global optimization during the last three decades, and the explosive growth of the computing capabilities show great promise towards addressing these issues.

1.2 STRUCTURE OF GLOBAL OPTIMIZATION MODELS

The nonconvex mathematical models that we study in this book from the deterministic global optimization point of view belong to three classes. The first class takes the form :

$$\begin{aligned} \min_{\boldsymbol{x}} \quad & f(\boldsymbol{x}) \\ s.t. \quad & \boldsymbol{h}(\boldsymbol{x}) = \mathbf{0} \\ & \boldsymbol{g}(\boldsymbol{x}) \leq \mathbf{0} \\ & \boldsymbol{x} \in X \subseteq \Re^n \end{aligned} \tag{1.1}$$

where f, $\boldsymbol{h}$ and $\boldsymbol{g}$ are the objective function, and the vectors of the equality and inequality constraints, respectively, and $\boldsymbol{x}$ is a vector of continuous variables of size n.

Within this first class of global optimization models (1.1) we have (i) the biconvex and/or bilinear mathematical structure that is the focus of our study in Part I; (ii) the signomial mathematical structure that arises in the generalized geometric programming and which we study in Part II; and (iii) the general mathematical structure of f, $\boldsymbol{h}$ and $\boldsymbol{g}$ belonging to $\mathcal{C}^2$, the set of twice continuously differentiable functions, that we study in Part III.

The second class of global optimization models (1.2) that we study in Part IV corresponds to mixed-integer nonlinear MINLP optimization problems that take the form :

$$\begin{aligned} \min_{\boldsymbol{x},\boldsymbol{y}} \quad & f(\boldsymbol{x},\boldsymbol{y}) \\ s.t. \quad & \boldsymbol{h}(\boldsymbol{x},\boldsymbol{y}) = \mathbf{0} \\ & \boldsymbol{g}(\boldsymbol{x},\boldsymbol{y}) \leq \mathbf{0} \\ & \boldsymbol{x} \in \boldsymbol{X} \subseteq \Re^n \\ & \boldsymbol{y} \in \boldsymbol{Y} \text{ integer,} \end{aligned} \tag{1.2}$$

where $\boldsymbol{x}$ represents a vector of n continuous variables and $\boldsymbol{y}$ is a vector of integer variables.

The third class of mathematical models (1.3) focuses on enclosing all solutions of a nonlinear system of equations subject to inequality constraints and variable bounds. This class of models, that we study in Part V, can be formu-

lated as:

$$\begin{aligned} h_j(\boldsymbol{x}) &= \mathbf{0}, \ j \in \mathcal{N}_E \\ g_k(\boldsymbol{x}) &\leq \mathbf{0}, \ k \in \mathcal{N}_I \\ \boldsymbol{x}^L &\leq \boldsymbol{x} \leq \boldsymbol{x}^U, \end{aligned} \tag{1.3}$$

where $\mathcal{N}_E$ is the set of equalities, $\mathcal{N}_I$ the set of inequality constraints, and $\boldsymbol{x}$ the vector of n continuous variables.

1.3 ILLUSTRATIVE APPLICATIONS OF GLOBAL OPTIMIZATION

In this section, we introduce a number of important applications of global optimization in the form of illustrative examples. These examples are revisited in Parts I-V of this book where we discuss the application of the deterministic global optimization methods. The applications introduced in this section and their respective mathematical classification are as follows :

- Pooling and Blending - Bilinear
- Minimization of Gibbs Free Energy - Biconvex
- Tangent Plane Stability Analysis - Biconvex
- Robust Stability Analysis - Signomial
- Structure Prediction in Clusters - Box Constrained Twice Differentiable NLP
- Protein Folding - Box Constrained Twice Differentiable NLP
- Parameter Estimation - General Constrained Twice Differentiable NLP
- Heat Exhanger Network Synthesis - Separable MINLP
- Pump Network Synthesis - General MINLP
- Multiple Steady States - All Solutions of Nonlinear Constrained Equations
- Homogeneous Azeotropes - All Solutions of Nonlinear Constrained Equations

1.3.1 Pooling Problems

Pooling problems represent an important component in refinery operations. In petrochemical processes it is necessary to model not only the product flows but also the properties of intermediate streams. These streams are usually combined in one or several tanks or pools. The pools are subsequently used in downstream processing or blending. The presence of these streams in the model introduces nonconvexities which are due to the interactions between the qualities of the input streams and the blended products.

The general formulation of pooling problems as studied by Ben-Tal *et al.* (1994), Floudas and Aggarwal (1990), Visweswaran and Floudas (1990), (1993), (1996b), Adjiman *et al.* (1998b) (see also references mentioned in these publications) take the form :

$$\max \quad -\sum_i \sum_l c_i x_{il} + \sum_l \sum_j d_j y_{lj} + \sum_i \sum_j (d_j - c_i) z_{ij}$$

$$\begin{aligned} s.t. \quad \sum_l x_{il} + \sum_j z_{ij} &\leq A_i \\ \sum_i x_{il} + \sum_j y_{lj} &= 0 \\ \sum_i x_{il} &\leq S_l \\ -\sum_i q_{ik} x_{il} + p_{lk} \sum_j y_{lj} &= 0 \\ \sum_l y_{lj} + \sum_i z_{ij} &\leq D_j \\ \sum_l (p_{lk} - Q_{jk}) y_{lj} + \sum_i (q_{ik} - Q_{jk}) z_{ij} &\leq 0. \end{aligned} \tag{1.4}$$

where the sets I, J, K, L represent the components i, the products j, the qualities k and the pools l, respectively. The variables are :

x_{il}	–	amount of component i allocated to pool l
y_{lj}	–	amount going from pool l to product j
z_{ij}	–	amount of component i going to product j
p_{lk}	–	level of quality k in pool l.

The parameters are :

A_i	–	Upper bounds for component availabilities
D_j	–	Upper bounds for product demands
S_l	–	Upper bounds for pool sizes
Q_{jk}	–	Upper bounds for product qualities
q_{ik}	–	Level of quality k in component i
c_i	–	Unit price of component i
d_j	–	Unit price of product j.

This mathematical model features nonconvexities in the form of bilinear terms in the constraints. These bilinear terms involve the products of the p_{lk} and y_{lj} variables. In Chapters 4, 7, and 13, we will revisit a variety of pooling problems and discuss the application of the deterministic global optimization approaches GOP and αBB.

1.3.2 Minimization of Gibbs Free Energy

The minimization of the Gibbs free energy problem is a fundamental problem in thermodynamics and more specifically in phase and chemical reaction equilibrium. It can be stated as :

Given a set of components $C = \{i\}$, the elements that constitute these components $E = \{e\}$, the set of phases $P = \{k\}$ consisting of vapor and liquid phases, denoted as P_V and P_L respectively, the liquid phase mol fractions $\boldsymbol{x} \equiv \{x_i\}$, and the vapor phase mol fractions $\boldsymbol{y} \equiv \{y_i\}$, we would like to determine the mol vector $\boldsymbol{n}$ that minimizes the value of the Gibbs free energy while also satisfying the appropriate material balance constraints.

For simultaneous phase and chemical equilibrium where reaction does occur, conservation of the constituent atoms must be satisfied:

$$\sum_{i \in C} \sum_{k \in P} a_{ei} n_i^k = b_e \quad \forall \quad e \in E \tag{1.5}$$

where a_{ei} represents the number of gram-atoms of element e in component i, and b_e the total number of gram-atoms of element e in the system.

For phase equilibrium without chemical reaction, conservation over the components need only hold:

$$\sum_{k \in P} n_i^k = n_i^T \quad \forall \quad i \in C \tag{1.6}$$

where n_i^T is the total number of moles of component i in the beginning mixture.

The material balance constraints for any system, reacting or non-reacting, will be written in the following general form:

$$\mathbf{A} \cdot \boldsymbol{n} - \mathbf{b} = 0 \tag{1.7}$$

where $\boldsymbol{n}$ represents the column vector of the component mol numbers, $\mathbf{A}$ is the appropriate matrix, and $\mathbf{b}$ is the column vector of the total amounts of elements or compounds in the system.

A physically realizable solution requires that

$$0 \leq n_i^k \leq n^T \qquad \forall \; i \in C \, , \; k \in P \tag{1.8}$$

where n^T is the total number of mols in the system.

The Gibbs free energy expression for the case of an ideal vapor phase and liquid phases modeled using the NRTL activity coefficient expression (Renon and Prausnitz, 1968) and after application of the theoretical properties developed by McDonald and Floudas (1994) (see Chapter 6) can be written as a combination of a convex part and a nonconvex part:

$$\min \hat{G}(\boldsymbol{n}) = \sum_{k \in P} \mathcal{C}_N^k + \sum_{i \in C} \sum_{k \in P_L} n_i^k \left\{ \sum_{j \in C} \frac{\mathcal{G}_{ij} \tau_{ij} n_j^k}{\sum\limits_{l \in C} \mathcal{G}_{lj} n_l^k} \right\} \tag{1.9}$$

where $\mathcal{C}_N^k$ is defined as follows:

$$\mathcal{C}_N^k = \sum_{i \in C} n_i^k \left\{ \frac{\Delta G_i^{k,f}}{RT} + \ln \frac{n_i^k}{\sum_{j \in C} n_j^k} \right\} \qquad \forall\, k \in P$$

and it is a convex function.

If the following new variables are introduced:

$$\Psi_i^k = \frac{n_i^k}{\sum_{j \in C} \mathcal{G}_{ji} n_j^k} \qquad \forall\ i \in C\ ,\ k \in P_L \tag{1.10}$$

then the transformed objective function becomes:

$$\min\ \hat{G}(\boldsymbol{\Psi}, \boldsymbol{n}) = \sum_{k \in P} \mathcal{C}^k + \sum_{i \in C} \sum_{k \in P_L} n_i^k \left\{ \sum_{j \in C} \mathcal{G}_{ij} \tau_{ij} \Psi_j^k \right\}. \tag{1.11}$$

Hence, the minimization of the Gibbs free energy problems using the NRTL equation takes the form :

$$\begin{aligned}
\min \quad & \hat{G}(\boldsymbol{\Psi}, \boldsymbol{n}) \\
s.t. \quad & \Psi_i^k \left\{ \sum_{j \in C} \mathcal{G}_{ji} n_j^k \right\} = n_i^k \qquad \forall\ i \in C\ ,\ k \in P_L \\
& \mathbf{A}\, \boldsymbol{n} - \mathbf{b} = \mathbf{0} \\
& 0 \leq n_i^k \leq n^T \quad \forall\, i \in C\ ,\ \forall\, k \in P.
\end{aligned}$$

This mathematical formulation features nonconvexities in the form of a biconvex objective function and a set of bilinear constraints. In Chapter 6, we will revisit this class of mathematical problems and discuss the theoretical, algorithmic issues and applications of the GOP approach.

1.3.3 Tangent Plane Stability Analysis

The tangent plane stability criterion is also a fundamental problem in thermodynamics and in particular in the phase and chemical reaction equilibrium. It focuses on establishing whether a postulated solution is thermodynamically stable with respect to perturbations in any or all of the phases. A necessary and sufficient condition for a postulated solution to be an equilibrium one is that the tangent plane distance function, denoted $\mathcal{F}(\boldsymbol{x})$, be nonnegative *for all possible phases represented in the system*, with $\mathcal{F}(\boldsymbol{x})$ defined as follows:

$$\mathcal{F}(\boldsymbol{x}) = \sum_{i \in C} x_i \left\{ \mu_i(\boldsymbol{x}) - \mu_i^0(\boldsymbol{z}) \right\} \tag{1.12}$$

where $\mu_i(\boldsymbol{x})$ and $\mu_i^0(\boldsymbol{z})$ represent the chemical potentials calculated at $\boldsymbol{x}$ and $\boldsymbol{z}$ respectively. The tangent plane distance function is defined as the distance between the Gibbs surface and the tangent plane constructed to this surface at z. If we define :

$$\begin{aligned} g(\boldsymbol{x}) &= \sum_{i\in C} x_i\mu_i(\boldsymbol{x}) && \text{Molar Gibbs surface (for a single phase),} \\ \mathcal{T}(\boldsymbol{x}) &= \sum_{i\in C} x_i\mu_i^0(\boldsymbol{z}) && \text{Tangent plane to this surface at } z, \\ \mathcal{F}(\boldsymbol{x}) &= g(\boldsymbol{x}) - \mathcal{T}(\boldsymbol{x}) && \text{Tangent plane distance function,} \end{aligned}$$

then the postulated solution corresponds to a global minimum of the Gibbs free energy as long as the tangent plane lies completely below the Gibbs surface.

The formulation for the tangent plane stability problem is :

$$\begin{aligned} \min \quad & \mathcal{F}(\boldsymbol{x}) = g(\boldsymbol{x}) - \sum_{i\in C} x_i\mu_i^0(\boldsymbol{z}) \\ s.t. \quad & \sum_{i\in C} x_i = 1 \\ & 0 \le x_i \le 1. \end{aligned}$$

Note that the tangent plane distance function is explicitly minimized. If a nonnegative global optimum solution is obtained, then the stability of the postulated solution is guaranteed.

Based on the analysis and properties proposed by McDonald and Floudas (1995b) for the NRTL equation, the objective function can be written as :

$$\mathcal{F}(\boldsymbol{x}) = \mathcal{C}(\boldsymbol{x}) + \sum_{i\in C} x_i \frac{\sum_{j\in C} \tau_{ij}\mathcal{G}_{ij}x_j}{\sum_{l\in C} \mathcal{G}_{lj}x_l}. \tag{1.13}$$

where

$$\mathcal{C}(\boldsymbol{x}) = \sum_{i\in C} x_i \left\{\Delta G_i^f + \ln x_i - \mu_i^0(\boldsymbol{z})\right\} \tag{1.14}$$

is a convex function.

By introducing new variables labeled $\mathcal{X}_i$,

$$\mathcal{X}_i = \frac{x_i}{\sum_{j\in C} \mathcal{G}_{ji}x_j} \quad \forall\, i \in C \tag{1.15}$$

results in the following tangent plane stability analysis problem for the NRTL activity coefficient equation :

$$\begin{aligned}
\min \quad & \mathcal{F}(\boldsymbol{x}) = \mathcal{C}(\boldsymbol{x}) + \sum_{i \in C} x_i \cdot \sum_{j \in C} \mathcal{G}_{ij} \tau_{ij} \mathcal{X}_j \\
s.t. \quad & \mathcal{X}_i \cdot \sum_{j \in C} \mathcal{G}_{ji} x_j = x_i \quad \forall\, i \in C \\
& \sum_{i \in C} x_i = 1 \ . \\
& 0 \leq x_i \leq 1 \quad \forall\, i \in C.
\end{aligned}$$

This mathematical model has nonconvexities in the form of a biconvex objective function and a set of bilinear constraints. In Chapter 6, we will discuss the theoretical and algorithmic issues of the GOP and its application to several phase equilibrium problems.

1.3.4 Robust Stability Analysis

Robust stability analysis of linear systems involves the identification of the largest possible region in the uncertain model parameter space for which the controller manages to attenuate any disturbances in the system. The stability of a feedback structure is determined by the roots of the closed loop characteristic equation:

$$\det\left(I + P(s, \boldsymbol{q}) C(s, \boldsymbol{q})\right) = 0$$

where $\boldsymbol{q}$ is the vector of the uncertain model parameters, and $P(s), C(s)$ are the transfer functions of the plant and controller respectively. After expanding the determinant we have the characteristic polynomial:

$$P(s, \boldsymbol{q}) = a_n(\boldsymbol{q}) s^n + a_{n-1}(\boldsymbol{q}) s^{n-1} + \cdots + a_1(\boldsymbol{q}) s + a_0(\boldsymbol{q}) = 0$$

where the coefficients $a_i(\boldsymbol{q}),\ i = 0, \ldots, n$ are typically multivariable polynomial functions. Real parameter uncertainty is typically expressed as positive and negative deviations of the real parameters from some nominal values.

Checking the stability of a particular system with characteristic equation $P(j\omega, \boldsymbol{q})$ involves the solution of the following nonconvex optimization problem (see Chapter 9 and Psarris and Floudas (1995)) :

$$\min_{q_i, k \geq 0, \omega \geq 0} \quad k \tag{1.16}$$

$$\begin{aligned}
\text{subject to} \quad Re\left[P(j\omega, \boldsymbol{q})\right] &= 0 \\
Im\left[P(j\omega, \boldsymbol{q})\right] &= 0 \\
q_i^N - \Delta q_i^- k \leq q_i &\leq q_i^N + \Delta q_i^+ k, \quad i = 1, \ldots, n
\end{aligned}$$

where $\boldsymbol{q}^N$ is a stable nominal point for the uncertain parameters and $\Delta\boldsymbol{q}^+, \Delta\boldsymbol{q}^-$ are the estimated bounds. Stability is ensured if and only if the optimal solution k is greater or equal to one.

This mathematical model features nonconvexities in the form of multivariable polynomial functions, known as signomials, that appear in the set of constraints. In Chapters 8 and 9, we will discuss a global optimization approach for optimization problems that involve signomials and we will present the theoretical and algorithmic issues along with several applications.

1.3.5 Structure Prediction in Clusters

Clusters are aggregates of atoms, ions, or molecules, sufficiently small that a significant proportion of these units is present on their surfaces. The structure prediction of microclusters is of primary importance in a variety of disciplines that range from thermodynamics to material science to catalysis to astrophysics. The cluster structure prediction problem can be stated as :

Given N particles interacting with two–body central forces, find their configuration(s) in the three-dimensional Euclidean space involving the global minimum total potential energy.

Assuming that (i) many–body and angle dependent interactions need not be taken into account, (ii) conformational entropy is not considered, and (iii) the particles are spherical and are of the same size, the structure prediction problem of a N–particle cluster can be written as:

$$\begin{aligned} \min_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}} V &= \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} v(r_{ij}) \qquad (1.17) \\ \text{where} \quad r_{ij} &= \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2} \\ \text{and} \quad & x_1 = y_1 = z_1 = y_2 = z_2 = z_3 = 0. \end{aligned}$$

where V is the total potential energy of the microcluster as the summation of all two-body interaction terms, $v(r_{ij})$ is the potential energy term corresponding to the interaction of particle i with particle j, and r_{ij} is the Euclidean distance between i and j.

Pair potentials that have been used in cluster studies include the following (Hoare, 1979):

Mie : $v(r) = (n-m)^{-1}[nr^{-m} - mr^{-n}]$,

Lennard–Jones : $v(r) = 4\epsilon\left\{\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6}\right\}$,

Morse : $v(r) = [1 - e^{a(1-r)}]^2 - 1$,

Gaussian : $v(r) = Ae^{-ar^2} - Be^{-br^2}$,

Born–Meyer : $v(r) = z^{\alpha}z^{\beta}/r + Ae^{-r/\rho}$, .

Due to the nonlinear form of the pairwise potentials the resulting mathematical model features a twice diffentiable nonlinear and nonconvex objective function subject to box constraints. This problem exhibits a very large number of local minima that grows significantly as the number of particles increases. We will revisit this class of problems in Chapter 14 where we will discuss a novel global optimization approach that exploits its mathematical structure and provides valid lower and upper bounds on the energy of the global minimum structure.

1.3.6 Structure Prediction in Protein Folding

Proteins are very complex molecules and this complexity is due to the intra- and inter-molecular interactions which define the native three-dimensional structure of the system, and subsequently its biological functionality. The structure prediction of proteins is a fundamental problem of utmost importance in computational chemistry and molecular biology (Leach (1991), Scheraga (1992), Pardalos *et al.* (1994), Vásquez *et al.* (1994), Pardalos *et al.* (1996), Dill and Chan (1997), Neumaier (1997b), Chan and Dill (1998), and Floudas *et al.* (1999b)).

In the internal coordinate space and employing an atomistic level mathematical model that establishes all the interactions (e.g., AMBER (Weiner *et al.*, 1984; Weiner *et al.*, 1986), CHARMM (Brooks *et al.*, 1983a), DISCOVER (Dauber-Osguthorpe *et al.*, 1988), ECEPP/3 (Némethy *et al.*, 1992)), the structure prediction problem in protein folding can be stated as :

$$
\begin{aligned}
\min \quad & E(\phi_i, \psi_i, \omega_i, \chi_i^k, \theta_j^N, \theta_j^C) \qquad (1.18)\\
\text{subject to} \quad & -\pi \le \phi_i \le \pi, \ i = 1, \ldots, N_{RES}\\
& -\pi \le \psi_i \le \pi, \ i = 1, \ldots, N_{RES}\\
& -\pi \le \omega_i \le \pi, \ i = 1, \ldots, N_{RES}\\
& -\pi \le \chi_i^k \le \pi, \ i = 1, \ldots, N_{RES}, \ k = 1, \ldots, K^i\\
& -\pi \le \theta_j^N \le \pi, \ j = 1, \ldots, J^N\\
& -\pi \le \theta_j^C \le \pi, \ j = 1, \ldots, J^C.
\end{aligned}
$$

where E represents the total potential energy function (using for instance ECEPP/3, see Chapter 16) and the free energy of solvation which can be

caclulated via methods based on the area accessible to the solvent or methods based on the volume accessible to the solvent; $i = 1, \ldots, N_{RES}$ is an indexed set describing the sequence of amino acid residues in the peptide chain; ϕ_i, ψ_i, ω_i, $i = 1, \ldots, N_{RES}$ are the dihedral angles along the backbone of this peptide; χ_i^k, $i = 1, \ldots, N_{RES}$, $k = 1, \ldots, K^i$ denote the dihedral angles of the side chains for the ith residue; θ_j^N, $j = 1, \ldots, J^N$ and θ_j^C, $j = 1, \ldots, J^C$ are the amino and carboxyl end group dihedral angles, respectively.

The resulting optimization problem for the structure prediction in protein folding is a box constrained nonlinear twice differentiable optimization problem where the nonconvexities are in the highly nonlinear form of the objective function. In Chapter 16, we will revisit the protein folding problem and discuss a novel global optimization framework based on the αBB approach for the determination of the global minimum energy structure, lower and upper bounds on the global minimum, as well as an ensemble of low energy conformers.

1.3.7 Parameter Estimation and Data Reconciliation

Nonlinear mathematical models which accurately predict physical phenomena are essential in engineering and applied science. The parameter estimation and data reconciliation problem is important in the design, synthesis, control and operations of chemical processes (Ricker (1984), Stewart *et al.* (1992), Kim *et al.* (1990)).

The general model consists of an implicit algebraic system of equations

$$\mathbf{f}(\theta, \mathbf{z}) \;=\; \mathbf{0} \tag{1.19}$$

where θ is a vector of p unknown parameters, $\mathbf{z}$ is a vector of n experimentally measured variables, and $\mathbf{f}$ represents the system of l algebraic functions. All experimentally measured variables are affected to some extent by error. The measurements are related to the true values through:

$$\mathbf{z}_\mu \;=\; \zeta_\mu \;+\; \mathbf{e}_\mu \qquad \mu \;=\; 1 \ldots m \tag{1.20}$$

where ζ_μ is vector of unknown true values of the experimentally measured variables, $\mathbf{z}_\mu$, at the μ data point, and $\mathbf{e}_\mu$ is the vector of additive error.

Based on the maxiumum likelihood approach and assuming that the errors in each experiment are known, equal, and independent, the parameter estimation and data reconciliation problem known as the *error-in-variables* formulation takes the form :

$$\min_{\hat{\mathbf{z}}_\mu\, \theta} \sum_{\mu=1}^{r} \sum_{i=1}^{m} \frac{(\hat{z}_{\mu,i} \;-\; z_{\mu,i})^2}{\sigma_i^2} \tag{1.21}$$

s.t.

$$\mathbf{f}(\hat{\mathbf{z}}_\mu\,, \theta) \;=\; \mathbf{0} \qquad \mu \;=\; 1 \ldots m.$$

where the optimization variables, $\hat{\mathbf{z}}_\mu$, are the fitted data variables that are approximations of the true values of the experimental data, ζ_μ, and σ_i is the standard deviation assigned to the ith variable in all experiments.

The resulting mathematical model of the parameter estimation and data reconciliation problem features a convex objective function while the nonconvexities are in the set of constraints that depend on the form of the nonlinear model. In Chapter 19, we will revisit the *error-in-variables* formulation and address it from the global optimization point of view.

1.3.8 Heat Exhanger Network Synthesis

The optimization-based approach in the area of *Process Synthesis* and more specifically for the synthesis of heat exchanger networks consists of (i) generating a superstructure of all alternatives of interest via graph theory, (ii) modeling the superstructure as a mixed-integer nonlinear optimization problem, and (iii) developing efficient algorithmic and solution strategies that extract the optimal structure from the set of alternatives embedded in the postulated superstructure (Floudas (1995)).

Given the set ST of temperature locations, the number of stages NS, the set HP of hot process streams and the set CP of cold process streams, employing the superstructure and formulation of Yee and Grossmann (1991), and using the Chen approximation for the logarithmic mean of the temperature difference (Chen, 1987), the general problem formulation takes the form :

$$
\begin{aligned}
\min \; & \sum_{i\in HP} C_{CU}Q_{CU,i} + \sum_{j\in CP} C_{HU}Q_{HU,j} \\
& + \sum_{i\in HP}\sum_{j\in CP}\sum_{k\in ST} CF_{ij}z_{ijk} + \sum_{i\in HP} CF_{i,CU}z_{CU,i} + \sum_{j\in CP} CF_{j,HU}z_{HU,j} \\
& + \sum_{i\in HP}\sum_{j\in CP}\sum_{k\in ST} \frac{C_{ij}Q_{ijk}}{U_{ij}[\Delta T_{ijk}\Delta T_{ijk+1}(\Delta T_{ijk}+\Delta T_{ijk+1})/2]^{\frac{1}{3}}} \\
& + \sum_{i\in HP} \frac{C_{A,i,CU}Q_{CU,i}}{U_{CU,i}[\Delta T_{CU,i}(T_{out,i}-T_{in,CU})(\Delta T_{CU,i}+T_{out,i}-T_{in,CU})/2]^{\frac{1}{3}}} \\
& + \sum_{j\in CP} \frac{C_{A,j,HU}Q_{HU,j}}{U_{HU,j}[\Delta T_{HU,j}(T_{in,HU}-T_{out,j})(\Delta T_{HU,j}+T_{in,HU}-T_{out,j})/2]^{\frac{1}{3}}}
\end{aligned}
$$

$$
\begin{aligned}
s.t. \quad (T_{in,i} - T_{out,i})\,Fcp_i &= \sum_{k\in ST}\sum_{j\in CP} Q_{ijk} + Q_{CU,i}, && \forall i \in HP \\
(T_{out,j} - T_{in,j})\,Fcp_j &= \sum_{k\in ST}\sum_{i\in HP} Q_{ijk} + Q_{HU,j}, && \forall j \in CP \\
(T_{i,k} - T_{i,k+1})\,Fcp_i &= \sum_{j\in CP} Q_{ijk}, && \forall k \in ST,\ \forall i \in HP \\
(T_{j,k} - T_{j,k+1})\,Fcp_j &= \sum_{i\in HP} Q_{ijk}, && \forall k \in ST,\ \forall j \in CP \\
T_{in,i} &= T_{i,1}, && \forall i \in HP \\
T_{in,j} &= T_{j,NS+1}, && \forall j \in CP \\
T_{i,k} &\geq T_{i,k+1}, && \forall k \in ST,\ \forall i \in HP \\
T_{j,k} &\geq T_{j,k+1}, && \forall k \in ST,\ \forall j \in CP \\
T_{out,i} &\leq T_{i,NS+1}, && \forall i \in HP \\
T_{out,j} &\geq T_{j,1}, && \forall j \in CP \\
(T_{i,NS+1} - T_{out,i})\,Fcp_i &= Q_{CU,i}, && \forall i \in HP \\
(T_{out,j} - T_{j,1})\,Fcp_j &= Q_{HU,j}, && \forall j \in CP
\end{aligned}
$$

$$
\begin{aligned}
Q_{ijk} - \Omega z_{ijk} \leq 0, \quad & \forall k \in ST,\ \forall i \in HP, \\
& \forall j \in CP \\
Q_{CU,i} - \Omega z_{CU,i} \leq 0, \quad & \forall i \in HP \\
Q_{HU,j} - \Omega z_{HU,j} \leq 0, \quad & \forall j \in CP \\
T_{i,k} - T_{j,k} + \Gamma(1 - z_{ijk}) \geq \Delta T_{ijk}, \quad & \forall k \in ST,\ \forall i \in HP, \\
& \forall j \in CP \\
T_{i,k+1} - T_{j,k+1} + \Gamma(1 - z_{ijk}) \geq \Delta T_{ijk+1}, \quad & \forall k \in ST,\ \forall i \in HP, \\
& \forall j \in CP \\
T_{i,NS+1} - T_{out,CU} + \Gamma(1 - z_{CU,i}) \geq \Delta T_{CU,i}, \quad & \forall i \in HP \\
T_{out,HU} - T_{j,1} + \Gamma(1 - z_{HU,j}) \geq \Delta T_{HU,j}, \quad & \forall j \in CP \\
\Delta T_{ijk} \geq 10, \quad & \forall k \in ST,\ \forall i \in HP, \\
& \forall j \in CP \\
z_{ijk}, z_{CU,i}, z_{HU,j} \in \{0,1\}, \quad & \forall k \in ST,\ \forall i \in HP, \\
& \forall j \in CP.
\end{aligned}
$$

where the continuous variables are : T_{ik}, the temperature of hot stream i at the hot end of stage k; T_{jk}, the temperature of cold stream j at the cold end of stage k; Q_{ijk}, the heat exchanged between hot stream i and cold stream j at temperature location k; $Q_{CU,i}$, the heat exchanged between hot stream i and the cold utility; $Q_{HU,j}$, the heat exchanged between cold stream j and the hot utility; ΔT_{ijk}, the temperature approach for the match of hot stream i and cold stream j at temperature location k; $\Delta T_{CU,i}$, the temperature approach for the match of hot stream i and the cold utility; $\Delta T_{HU,j}$, the temperature approach for the match of cold stream j and the hot utility.

The binary variables are : z_{ijk}, for the existence of a match between hot stream i and cold stream j at temperature location k; $z_{CU,i}$, for the existence of a match between hot stream i and the cold utility; $z_{HU,j}$, for the existence of a match between cold stream j and the hot utility.

The parameters are : T_{in}, the inlet temperature of a stream; T_{out}, the outlet temperature; Fcp, the heat capacity flowrate of a stream; Ω, the upper bound on heat exchange; Γ, the upper bound on the temperature difference.

The resulting mathematical model corresponds to a mixed-integer nonlinear optimization problem where the binary variables appear linearly and separably from the continuous variables in both the objective function and constraints. All the constraints are linear while the nonconvexities appear in the objective function. In Chapter 21, we will revisit this class of nonconvex MINLP problems and discuss the SMIN-αBB global optimization approach and its application to the heat exchanger network synthesis problems.

1.3.9 Pump Network Synthesis

The synthesis of pump networks involves the postulation of a multi-level superstructure and its formulation as a general MINLP model (Westerlund *et al.* (1994)). The objective is to identify the least costly configuration of centrifugal pumps that achieves a pre-specified pressure rise based on a given total flowrate.

The structural decisions for an L level superstructure are represented by a number of discrete variables. The binary variables z_i, $i = 1, \ldots, L$, denote the existence of level i. The integer variables Np_i, $i = 1, \ldots, L$, denote the number of parallel lines at level i ($Np_i \in \{0, \ldots, NP\}$, where NP is the maximum number of parallel lines). The integer variables Ns_i, $i = 1, \ldots, L$, denote the number of pumps in series at level i ($Ns_i \in \{0, \ldots, NS\}$, where NS is the maximum number of pumps in series). The key continuous variables are the fraction of total flow going to level i, x_i, the flowrate on each line at level i, $\dot{v}_i$, the rotation speed of all pumps on level i, ω_i, the power requirements at level i, P_i and the pressure rise at level i, Δp_i.

The optimization model for a three-level (i.e., $L = 3$) superstructure, that minimizes the annualized cost takes the form :

$$
\begin{aligned}
& \min \sum_{i=1}^{3} (C_i + C_i' P_i) Np_i Ns_i z_i \\
s.t. \quad & P_1 - 19.9(\tfrac{\omega_1}{\omega_{max}})^3 - 0.1610(\tfrac{\omega_1}{\omega_{max}})^2 \dot{v}_1 + 0.000561(\tfrac{\omega_1}{\omega_{max}}) \dot{v}_1^2 = 0 \\
& P_2 - 1.21(\tfrac{\omega_2}{\omega_{max}})^3 - 0.0644(\tfrac{\omega_2}{\omega_{max}})^2 \dot{v}_2 + 0.000564(\tfrac{\omega_2}{\omega_{max}}) \dot{v}_2^2 = 0 \\
& P_3 - 6.52(\tfrac{\omega_3}{\omega_{max}})^3 - 0.1020(\tfrac{\omega_3}{\omega_{max}})^2 \dot{v}_3 + 0.000232(\tfrac{\omega_3}{\omega_{max}}) \dot{v}_3^2 = 0 \\
& \Delta p_1 - 629(\tfrac{\omega_1}{\omega_{max}})^2 - 0.696(\tfrac{\omega_1}{\omega_{max}}) \dot{v}_1 + 0.0116 \dot{v}_1^2 = 0 \\
& \Delta p_2 - 215(\tfrac{\omega_2}{\omega_{max}})^2 - 2.950(\tfrac{\omega_2}{\omega_{max}}) \dot{v}_2 + 0.0115 \dot{v}_2^2 = 0 \\
& \Delta p_3 - 361(\tfrac{\omega_3}{\omega_{max}})^2 - 0.530(\tfrac{\omega_3}{\omega_{max}}) \dot{v}_3 + 0.00946 \dot{v}_3^2 = 0 \\
& x_1 + x_2 + x_3 = 1 \\
& \omega_i - \omega_{max} \leq 0, \; i = 1, 2, 3 \\
& \dot{v}_i - \tfrac{x_i}{Np_i} V_{tot} = 0, \; i = 1, 2, 3 \\
& \Delta P_{tot} - \Delta p_i Ns_i = 0, \; i = 1, 2, 3 \\
& Np_i = \{0, 1, 2, 3\}, \; i = 1, 2, 3 \\
& Ns_i = \{0, 1, 2, 3\}, \; i = 1, 2, 3 \\
& z_i = \{0, 1\}, \; i = 1, 2, 3.
\end{aligned}
\tag{1.22}
$$

This mathematical formulation is mixed integer nonlinear and does not exhibit any particular structure. It is nonconvex in both the continuous and integer variables. The nonconvexities appear in the objective function and con-

straints, and there exist integer and binary variables. In Chapter 22, we will revisit this general class of nonconvex MINLPs and discuss the GMIN-αBB global optimization approach along with its application to the pump network synthesis and trim loss minimization problems.

1.3.10 Multiple Steady States in Reactors

The enclosure of all solutions of nonlinear constrained systems of equations is regarded as a very challenging task in applied mathematics, engineering and sciences. A class of such problems is the location of all steady states in chemical reactor systems.

As an illustration, we consider here a model of two continuous, non-adiabatic, stirred tank reactors in series at steady state with recycle and an exothermic, first order, irreversible reaction (Kubicek *et al.*, 1980). The modelling equations of this system involve products between linear and exponential terms and are:

$$\begin{aligned}
(1-R)(1-y_1)\exp\left(\frac{10\phi_1}{1+10\phi_1/\gamma}\right) - y_1 &= 0 \\
(1-R)D(1-y_1)\exp\left(\frac{10\phi_1}{1+10\phi_1/\gamma}\right) - 10(1+\beta_1)\phi_1 + \beta_1\phi_{c1} &= 0 \\
y_{11} - y_2 + (1-R)(1-y_2)\exp\left(\frac{10\phi_2}{1+10\phi_2/\gamma}\right) &= 0 \\
10\phi_1 - 10(1+\beta_2)\phi_2 + (1-R)D(1-y_2)\exp\left(\frac{10\phi_2}{1+10\phi_2/\gamma}\right) + \beta_2\phi_{c2} &= 0,
\end{aligned}$$

where the variables are:

$$\begin{aligned}
y_i &= \text{Reactant conversion in reactor } i{=}1{,}2 \\
\phi_i &= \text{Dimensionless temperature in reactor } i{=}1{,}2.
\end{aligned}$$

and the parameters are:

$$\begin{aligned}
\gamma &= 1000, \text{ Dimensionless activation energy} \\
D &= 22, \text{ Dimensionless adiabatic temperature rise} \\
\phi_{c1} = \phi_{c2} &= 0, \text{ Dimensionless coolant temperatures} \\
\beta_1 = \beta_2 &= 2, \text{ Dimensionless heat transfer coefficient in reactors 1,2} \\
0 \le R &\le 1, \text{ Recycle ratio.}
\end{aligned}$$

For different values of the recycle ratio R, starting with R=0.935 and ending at R=0.995 with 0.005 increments the above system of nonlinear constrained equations exhibits one, three, five, or seven solutions.

This mathematical model corresponds to a nonlinear constrained system of equations which is nonconvex. In Chapter 23, we will revisit this class of problems, discuss a deterministic global optimization approach for the enclosure of all solutions and apply it to several challenging problems.

1.3.11 Enclosure of All Homogeneous Azeotropes

The ability to predict whether a given mixture will form one or more homogeneous, heterogeneous and reactive azeotropes and to calculate the conditions and compositions of each azeotrope is essential in modeling separation processes.

Considering here the case of homogeneous azeotropes (i.e., one liquid phase), there are three thermodynamic conditions which a system must meet in order for an homogeneous azeotrope to exist. These conditions are: (i) equilibrium, (ii) the composition of vapor phase must be identical to the overall composition of the liquid phase, and (iii) the mole fractions of the components in each phase must sum to unity and must be non-negative.

The equilibrium condition requires that the chemical potential of each component must be the same in all phases. Since an homogeneous azeotropic system contains a vapor phase (V), and only one liquid phase (L), this condition can be written:

$$\mu_i^V = \mu_i^L \qquad \forall\, i \in N \tag{1.23}$$

where μ_i^V and μ_i^L represent the chemical potential of component i in the vapor and liquid phases. From the definition of the fugacity of component i in a mixture, $\hat{f}_i$,

$$\hat{f}_i^V = \hat{f}_i^L \qquad \forall\, i \in N \tag{1.24}$$

hence,

$$y_i \hat{\phi}_i^V P = x_i \gamma_i^L f_i^L. \tag{1.25}$$

The symbol $\hat{\phi}_i^V$ represents the mixture fugacity coefficient of component i in the vapor phase. For the liquid phase, γ_i^L is the activity coefficient, and f_i^L is the fugacity of component i in the liquid phase. Assuming that the vapor phase can be modeled as an ideal gas, we have :

$$\frac{y_i}{x_i} = \frac{\gamma_i^L P_i^{sat}}{P}. \tag{1.26}$$

Note that all of the nonidealities of the system are confined to the activity coefficient term. For instance, employing the Wilson activity coefficient (Wilson, 1964) equation we have :

$$\ln \gamma_i = 1 - \ln\left(\sum_{j \in N} x_j \Lambda_{ij}\right) - \sum_{j \in N} \frac{x_j \Lambda_{ji}}{\sum\limits_{k \in N} x_k \Lambda_{jk}} \tag{1.27}$$

where Λ_{ij} is the nonsymmetric binary interaction parameter between components i and j with $\Lambda_{ii} = 1$.

The saturated vapor pressure is calculated using the Antoine equation:

$$\ln P_i^{sat} = a_i - \frac{b_i}{T + c_i} \tag{1.28}$$

where a_i, b_i, and c_i are constant parameters with a_i and b_i being positive, while c_i may be positive or negative but $\mid c_i \mid < T$.
Condition (ii) takes the form:

$$y_i = x_i \qquad \forall\, i \in N. \tag{1.29}$$

Condition (iii) requires that the mole fractions in each phase sum to unity and have values between 0 and 1.

$$\begin{aligned} \sum_{i \in N} y_i = \sum_{i \in N} x_i = 1 \\ 0 \leq y_i, x_i \leq 1 \qquad \forall\, i \in N. \end{aligned} \tag{1.30}$$

This mathematical model corresponds to a nonlinear constrained system of equations that exhibits multiplicity of solutions and each solution is an homogeneous azeotrope. In Chapter 24, we will revisit the location of all homogeneous azeotropes problem and discuss the application of a deterministic global optimization approach for the enclosure of all homogeneous azeotropes.

1.4 DETERMINISTIC GLOBAL OPTIMIZATION : APPROACHES AND CLASSES OF PROBLEMS

In the last three decades and to a larger extent in the last fifteen years, there has been a lot of interest in theoretical and computational investigations of challenging global optimization problems. This has resulted in the development and application of novel deterministic global optimization methods to important problems in science, applied science, and engineering. Exciting and intriguing theoretical findings accompanied by algorithmic developments have taken place not only for special classes of nonconvex problems but also for general constrained nonlinear problems, as well as for mixed-integer nonlinear problems.

The deterministic global optimization approaches and the classes of global optimization problems can be identified as :

Deterministic Global Optimization Approaches :

Branch and Bound methods

Cutting Plane methods

Primal-Dual Decomposition methods

Outer Approximation methods

Inner Approximation methods

Difference of Convex (D.C.) and Reverse Convex methods

Reformulation-Linearization methods

Lipschitzian methods

Trajectory and Homotopy methods

Interval Analysis methods

Classes of Global Optimization Problems :

Concave Minimization problems

D.C. and Reverse Convex problems

Lipschitzian Optimization problems

Quadratic Optimization problems

Bi-convex Optimization problems

Complementarity problems

Multiplicative Programming problems

Bilevel and Multi-level Optimization problems

Fractional Programming problems

Generalized Geometric Programming problems

Twice Continuously Differentiable, C^2, NLP problems

All Solutions of Nonlinear Constrained Systems of Equations problems

Mixed-Integer Nonlinear Optimization problems

Max-Min Optimization problems

Integer Programming problems

Network Optimization problems.

1.5 COMPLEXITY ANALYSIS

In this section, we present an outline of the basic concepts that are used in complexity theory. For a detailed discussion the reader is directed to the books by Garey and Johnson (1979) and Vavasis (1991).

The complexity analysis focuses on the asymptotic study of *problems* and *algorithms*, and therefore the complexity results correspond to *worst case* behavior of such problems and algorithms. Key questions that the complexity analysis addresses include :

Q1 : How difficult is it to solve a particular class of optimization problems?

Q2 : How efficient is an algorithm as the size of the variable space increases?

Q3 : What classes of optimization problems can be solved with efficient algorithms?

Q4 : Are there classes of optimization problems for which there do not exist efficient algorithms?

1.5.1 Basic Definitions

The following elementary definitions form the basis for the classification of the different problems and classes of algorithms.

Definition 1.5.1 (Problem) *Let the sets I and O be input and output sets of strings of symbols, respectively, and the set I be defined by a simple syntactic property. Then, a problem is defined as a mapping $F : I \longrightarrow O$.*

Illustration 1.5.1 As an illustration, we consider the following quadratic programming problem :

$$\begin{aligned} \min_{\boldsymbol{x}} \quad & \boldsymbol{c}^T\boldsymbol{x} + 0.5\boldsymbol{x}^T\mathbf{Q}\boldsymbol{x} \\ s.t. \quad & \boldsymbol{A}\boldsymbol{x} \leq \mathbf{b} \end{aligned}$$

The input set I consists of the quadruples $(\mathbf{Q}, \boldsymbol{c}, \boldsymbol{A}, \mathbf{b})$ which are written out in digits. The output set O consists of vectors written out as rational numbers. Then, F maps the quadruple $(\mathbf{Q}, \boldsymbol{c}, \boldsymbol{A}, \mathbf{b})$ to the solution $\boldsymbol{x}^*$ that solves the quadratic programming problem.

Definition 1.5.2 (Length of the Input) *The length of the input is defined as the total number of symbols that represent the input.*

Illustration 1.5.2 For example, the input $(\mathbf{Q}, \boldsymbol{c}, \boldsymbol{A}, \mathbf{b})$ for the quadratic programming problem consists of matrices and vectors of rational numbers, and the length of the input is the total number of digits in the numerators and denominators of all the rational numbers.

The Turing machine is a theoretical model of computation that has the following important elements : (i) it manipulates symbols that come from a finite set (e.g., the digits $0, \ldots, 9$; punctuation marks), (ii) it has an unbounded number of memory cells and each cell can hold one symbol, and (iii) it is controlled by a program of finite length. Considering the quadratic programming problem defined above, a Turing machine computes the function F, that is, it receives as input the quadruple $(\mathbf{Q}, \boldsymbol{c}, \boldsymbol{A}, \mathbf{b})$ and solves the quadratic programming problem returning the correct $\boldsymbol{x}^*$ as the output.

Definition 1.5.3 (Decision Problem) *A decision problem is a problem in which the output set O consists of only two elements, that is $O = \{\mathbf{yes}, \mathbf{no}\}$.*

Illustration 1.5.3 An example of a decision problem is when a feasible or infeasible solution is sought.

Definition 1.5.4 (Polynomial-time Algorithm) *An algorithm is said to be polynomial-time if the number of steps (i.e., the number of steps required by a Turing machine) is bounded by a polynomial in the length of the input.*

Definition 1.5.5 (P-class of problems) *The decision problems for which there exists a Turing machine that solves them such that its running time is*

bounded by a polynomial in the length of the input define the **P**-*class of problems.*

Illustration 1.5.4 The class of linear programming problems with rational numbers is in **P** when posed as a decision problem.

Remark 1.5.1 The **P**-class of problems are *easy* to solve (i.e., they can be solved efficiently in polynomial time).

A decision problem which is not in **P** is often considered as intractable, meaning that there does not exist an asymptotically efficient algorithm for it. The theory of **NP**-completeness forms the basis for such a classification.

Definition 1.5.6 (NP-class of problems) *A decision problem* $F : I \longrightarrow \{\mathbf{yes}, \mathbf{no}\}$ *is in* **NP** *if there exists a set* C *of strings defined by some simple syntactic property, and another decision problem* $G : I \times C \longrightarrow \{\mathbf{yes}, \mathbf{no}\}$ *such that :*

(a) G *is in* **P**, *and*

(b.1) *If* $F(\boldsymbol{x}) = \mathbf{yes}$, *then there exists at least one string* $\boldsymbol{y} \in C$ *such that* $\boldsymbol{y}$ *has length at most polynomial in* $\boldsymbol{x}$, *and such that* $G(\boldsymbol{x}, \boldsymbol{y}) = \mathbf{yes}$. *This string* $\boldsymbol{y} \in C$ *is denoted as a certificate of* $\boldsymbol{x}$.

(b.2) *If* $F(\boldsymbol{x}) = \mathbf{no}$, *then we have* $G(\boldsymbol{x}, \boldsymbol{y}) = \mathbf{no}$ *for all* $\boldsymbol{y} \in C$.

Remark 1.5.2 A decision problem which is in **NP** has all its **yes** instances easily certified. The **NP**-class of problems represents the class for which it is easy to certify the correctness of a solution that can be obtained through a tedious procedure. Also, note that $\mathbf{P} \subset \mathbf{NP}$.

Definition 1.5.7 (NP-complete class of problems) *A decision problem* $F : I \longrightarrow \{\mathbf{yes}, \mathbf{no}\}$ *in* **NP** *is defined as* **NP** $-$ *complete if every problem in* **NP** *can be reduced in polynomial time to* F.

Remark 1.5.3 The **NP** $-$ *complete* problem is the most difficult in **NP** since every problem in **NP** can be reduced efficiently to it.

Definition 1.5.8 (NP-hard problem) *A decision problem* $F : I \longrightarrow \{\mathbf{yes}, \mathbf{no}\}$ *is defined as* **NP** $-$ *hard if every problem in* **NP** *can be reduced to* F.

Remark 1.5.4 Note that the **NP** $-$ *hardness* condition is weaker than the **NP** $-$ *completeness* condition. This is because F does not need to be in **NP**.

1.5.2 Complexity of Local Optimization

It is interesting to study the local optimization algorithms, that attempt to compute improved solutions by perturbations in the neighborhood of the current solution, from the complexity point of view. This is especially true since

computing locally optimal solutions is considered easier than computing globally optimal solutions. The complexity results are summarized :

(i) The problem of checking local optimality for a feasible point is **NP**-hard.

(ii) The problem of checking whether a local minimum is strict is **NP**-hard.

1.5.3 Complexity of Global Optimization

General nonconvex optimization problems are usually **NP**-hard or **NP**-complete (Murty and Kabadi (1987), Pardalos and Schnitger (1988)). In the sequel, we outline first the complexity results of nonconvex optimization problems that are **NP**-hard or **NP**-complete, and second we present nonconvex optimization problems that can be solved in polynomial time.

(i) The quadratic programming problem :

$$\begin{aligned} \min_{\boldsymbol{x}} \quad & \boldsymbol{c}^T\boldsymbol{x} + 0.5\boldsymbol{x}^T\mathbf{Q}\boldsymbol{x} \\ s.t. \quad & \boldsymbol{A}\boldsymbol{x} \leq \mathbf{b} \end{aligned}$$

is **NP**-hard.

(ii) The box constrained quadratic problem :

$$\begin{aligned} \min_{\boldsymbol{x}} \quad & \boldsymbol{c}^T\boldsymbol{x} + 0.5\boldsymbol{x}^T\mathbf{Q}\boldsymbol{x} \\ s.t. \quad & \mathbf{0} \leq \boldsymbol{x} \leq \mathbf{1} \end{aligned}$$

is **NP**-hard.

(iii) The simplex constrained quadratic problem :

$$\begin{aligned} \min_{\boldsymbol{x}} \quad & \boldsymbol{c}^T\boldsymbol{x} + 0.5\boldsymbol{x}^T\mathbf{Q}\boldsymbol{x} \\ s.t. \quad & x_1 + \ldots + x_n = 1 \\ & \boldsymbol{x} \geq \mathbf{0} \end{aligned}$$

is **NP**-hard.

(iv) A quadratic problem with one negative eigenvalue is **NP**-hard.

(v) The quadratic knapsack problem :

$$\begin{aligned} \min_{\boldsymbol{x}} \quad & \sum_{i=1}^{n} b_i x_i + c_i x_i^2 \\ s.t. \quad & a_1 x_1 + \ldots + a_n x_n = t \\ & l_i \leq x_i \leq u_i, \quad i = 1, \ldots, n \end{aligned}$$

is **NP**-hard.

(vi) The quadratic programming problem posed as the decision problem $F : I \longrightarrow \{\mathbf{yes}, \mathbf{no}\}$ where the input set I is a 5-tuple $(\mathbf{Q}, \boldsymbol{c}, \boldsymbol{A}, \mathbf{b}, t)$, t is a rational number, the output set is **yes** if there is a feasible point to $\mathbf{A}\boldsymbol{x} \leq \mathbf{b}$ such that the objective function achieves a value of t or smaller, and it is **no** otherwise, is **NP**-complete.

(vii) The satisfiability problem is **NP**-complete.

(viii) The traveling salesman problem is **NP**-complete.

(ix) The maximum clique problem is **NP**-complete.

It should be noted that there are nonconvex optimization problems that can be solved in polynomial time. In the following we present two instances of such problems :

(i) The fractional linear programming problem

$$\begin{aligned} \min_{\boldsymbol{x}} \quad & \frac{\boldsymbol{c}^T\boldsymbol{x} + \mu)}{(\boldsymbol{d}^T\boldsymbol{x} + \lambda)} \\ s.t. \quad & \boldsymbol{A}\boldsymbol{x} \leq \mathbf{b} \end{aligned}$$

where the denominator does not change sign, can be solved in polynomial time.

(ii) The quadratic programming problem subject to a sphere constraint :

$$\begin{aligned} \min_{\boldsymbol{x}} \quad & \boldsymbol{c}^T\boldsymbol{x} + 0.5\boldsymbol{x}^T\mathbf{Q}\boldsymbol{x} \\ s.t. \quad & \boldsymbol{x}^T\boldsymbol{x} \leq 1 \end{aligned}$$

can be solved in polynomial time.

Remark 1.5.5 The problem of minimizing a quadratic function subject to a sphere arises in many general purpose optimization algorithms and it is denoted as the *model trust region problem.*

1.6 RESEARCH AREAS AND IMPORTANT APPLICATIONS

Global optimization applications are widespread in all disciplines of science, applied science, and engineering. The research areas include :

Research Areas

Computational Chemistry

Molecular Biology

Economics

Thermodynamics

Chemical Engineering Process Operations
Chemical Engineering Process Design
Chemical Engineering Process Synthesis
VLSI Design
Structural Optimization
Engineering Design
Nuclear and Mechanical Design
Production Management
Transportation and Communications Network Design
Logistics
Finance
Facility Location
Water Resources Management
Design under Uncertainty
Packing
Statistics
Parameter Estimation
Data Reconciliation

The important applications of global optimization include :

Important Applications

Fixed Charge Linear and Nonlinear Problems in Plant Location
Fixed Charge Linear and Nonlinear Problems in Production Lot-Sizing
Economies of Scale Problems
Portfolio Selection
Multicommodity Network Flow Problems
Optimal Design of Trusses
Robust Stability of Control Systems Under Parametric Uncertainty
Location-Allocation of Wells and Platforms
Air Traffic Control
Economic Equilibria and Competition
Phase and Chemical Equilibrium
Structure Prediction of Atomic and Molecular Clusters
Structure Prediction of Molecules
Protein Folding

Peptide Docking

Structure Refinement based on NMR and Crystallographic data

Location of First and Higher Order Transition States

Feasibility Test and Flexibility Index problems

Design under Uncertainty of Batch Processes

Parameter Estimation of Nonlinear Algebraic and Dynamic Models

Optimal Control of Nonlinear Processes

Pooling and Blending problems

Design of Nonideal Separations

Location of Homogeneous, Heterogeneous and Reactive Azeotropes

Synthesis of Heat Exchanger Networks

Synthesis of Complex Reactor Networks

Synthesis of Distillation Systems

Synthesis of Pump Network Configurations

Chip Layout and Compaction Problems

Process Layout Problems

Trim Loss Problems

Fabric and Metal Nesting Problems.

1.7 PUBLICATIONS AND CONFERENCES

The significant growth in the area of global optimization especially during the last fifteen years is evidenced not only by the remarkable increase of journal publications across all disciplines, but also by the emergence of (i) the *Journal of Global Optimization* which started in 1990, (ii) the book series on *Nonconvex Optimization and Its Applications* which started in 1994, (iii) the impressive list of textbooks and monographs that have been published, and (iv) five international conferences held in 1990, 1991, 1995, and 1999.

A representative list of textbooks and monographs focused on global optimization includes :

Textbooks and Monographs

Interval Analysis, Moore (1966)

Towards Global Optimization I, Dixon and Szego (1975)

Towards Global Optimization II, Dixon and Szego (1978)

Methods and Applications of Interval Analysis, Moore (1979)

Computer Methods for the Range of Functions, Ratschek and Rokne (1984)

Constrained Global Optimization : Algorithms and Applications, Pardalos and Rosen (1987)

New Computer Methods for Global Optimization, Ratschek and Rokne (1988)

Global Optimization, Torn and Zilinskas (1989)

Global Optimization : Deterministic Approaches, Horst and Tuy (1996), (1993), (1990)

A Collection of Test Problems in Constrained Global Optimization Algorithms, Floudas and Pardalos (1990)

Interval Methods for Systems of Equations, Neumaier (1990)

Global Optimization Using Interval Analysis, Hansen (1992)

Recent Advances In Global Optimization, Floudas and Pardalos (1992)

Handbook of Global Optimization, Horst and Pardalos (1994)

Introduction to Global Optimization, Horst *et al.* (1995)

Minimax and Applications, Du and Pardalos (1995)

Advances in Multicriteria Analysis, Pardalos *et al.* (1995)

Global Optimization in Action - Continuous and Lipschitz Optimization, Pinter (1995)

State of the Art in Global Optimization : Computational Methods and Applications, Floudas and Pardalos (1996)

Global Optimization In Engineering Design, Grossmann (1996)

Rigorous Global Search : Continuous Problems, Kearfott (1996)

Optimization on Low Rank Nonconvex Structures, Konno *et al.* (1997)

Bayesian Heuristic Approach to Discrete and Global Optimization, Mockus *et al.* (1996)

Developments in Global Optimization, Bomze *et al.* (1997)

Multilevel Optimization : Algorithms and Applications, Migdalas *et al.* (1997)

Nonconvex Optimization in Mechanics, Mistakidis and Stavroulakis (1997)

Convex Analysis and Global Optimization, Tuy (1998)

Nondifferentiable Optimization and Polynomial Problems, Shor (1998)

Generalized Convexity, Generalized Monotonicity : Recent Results, Crouzeix *et al.* (1998)

Nonsmooth Approach to Optimization Problems with Equilibrium Constraints, Outrata *et al.* (1998)

Handbook of Test Problems in Local and Global Optimization, Floudas *et al.* (1999)

The aforementioned textbooks and monographs were complemented with five international conferences with focal themes in the area of Global Optimization. These took place in the 1990's and are :

Global Optimization Conferences

Workshop on Global Optimization, Sopron, Hungary, December 1990, Organizers : IIASA.

Recent Advances In Global Optimization, Princeton University, Princeton, N.J. USA, May 1991, Organizers : C. A. Floudas and P. M. Pardalos.

State of the Art In Global Optimization : Computational Methods and Applications, Princeton University, Princeton, N.J. USA, May 1995, Organizers : C. A. Floudas and P. M. Pardalos.

Workshop on Global Optimization, Szeged, Hungary, December 1995, Organizers : I. M. Bomze, T. Csendes, R. Horst and P. M. Pardalos.

Optimization In Computational Chemistry and Molecular Biology : Local and Global Approaches, Princeton University, Princeton, N.J. USA, May 1999, Organizers : C. A. Floudas and P. M. Pardalos.

1.8 CHEMICAL ENGINEERING CONTRIBUTIONS

A significant effort has been expended in the last four decades toward theoretical and algorithmic studies of applications that arise in Chemical Engineering Process Design, Process Synthesis, Process Control, as well as in Computational Chemistry and Molecular Biology. In the last decade we have experienced a dramatic growth of interest in Chemical Engineering for new methods of global optimization and their application to important engineering, as well as computational chemistry and molecular biology problems.

Contributions from the chemical engineering community to the area of global optimization can be traced to the work of Stephanopoulos and Westerberg (1975), and Westerberg and Shah (1978). Renewed interest in seeking global solutions was motivated from the work of Floudas *et al.* (1989). The first exact primal-dual global optimization approach was proposed by Floudas and Visweswaran (1990), (1993) and its features were explored for quadratically constrained and polynomial problems in the work of Visweswaran and Floudas (1990), (1992), (1993). Swaney (1990) proposed a branch and bound global optimization approach and more recently Quesada and Grossmann (1993), (1995) combined convex underestimators in a branch and bound framework for linear fractional and bilinear programs. Manousiouthakis and Sourlas (1992) proposed a reformulation to a series of reverse convex problems, and Tsirukis and Reklaitis (1993) proposed a feature extraction algorithm for constrained global optimization. Maranas and Floudas (1992), (1993), (1994a), (1994b) proposed a novel branch and bound method combined with a difference of convex functions transformation for the global optimization of molecular conformation problems that arise in computational chemistry. A very important theoretical advance has been made by Liu and Floudas (1993), (1995), (1996) who showed that the GOP can be applied to very general classes of NLPs. This result is very significant because it extends the classes of mathematical problems on which the GOP can be applied to nonlinear objective function and constraints that are twice-continuously differentiable. Vaidyanathan

and El-Halwagi (1994) proposed an interval analysis based global optimization method. In a series of papers McDonald and Floudas (1994), (1995a), (1995b), (1995c) addressed the fundamental problems of (i) minimization of the Gibbs free energy and (ii) the tangent plane stability criterion that arise in phase and chemical reaction equilibrium as global optimization problems for the first time. They proposed decomposition based approaches for biconvex problems that result from the use of the NRTL equation, and branch and bound approaches for the UNIQUAC, UNIFAC, ASOG, and TK-Wilson equations. Psarris and Floudas (1995) proposed a global optimization approach based on the GOP for the robust stability of control systems under real parametric uncertainty. Ryoo and Sahinidis (1995) suggested the application of reduction tests within the framework of branch and bound methods and they proposed a branch and reduce global optimization approach (Ryoo and Sahinidis (1996)). Dorneigh and Sahinidis (1995) proposed a global optimization approach for the chip layout and compaction problem. Androulakis *et al.* (1995) proposed the global optimization method αBB which addresses general continuous optimization problems with nonconvexities in the objective function and/or constraints. This approach classifies the nonconvexities as special structure (e.g., bilinear, signomial, univariate) or generic structure and is based on convex relaxations and a branch and bound framework. Maranas and Floudas (1995) proposed a new approach for enclosing all ϵ-feasible solutions of nonlinearly constrained systems of equations. This approach transforms the problem into a min-max form and corresponds to enclosing all multiple global optima via the αBB global optimization approach.

The books of Floudas and Pardalos (1996) and Grossmann (1996) contain a number of recent chemical engineering contributions, and the review paper of Floudas (1997) presents an overview of these recent chemical engineering contributions and the advances in global optimization based on the decomposition approach GOP, the generalized geometric programming, and the αBB approach.

Some of the aforementioned chemical engineering contributions in 1996 are briefly discussed in the following. Adjiman and Floudas (1996) proposed novel approaches for the rigorous determination of the α parameters that are employed in the αBB global optimization approach. These methods are based on a modified Kharitonov theorem for interval polynomials, interval anlysis of the Hessian matrices and on calculations of rigorous bounds on the minimum eigenvalue for general twice differentiable problems. Staus *et al.* (1996) formulated the combined adaptive controller design and estimation problem as a nonconvex problem with convex objective and bilinear constraints, and proposed a branch and bound global optimization method which is based on the McCormick underestimators. Visweswaran *et al.* (1996) addressed bilevel linear and quadratic programming problems to global optimality by employing the basic principles of the GOP and developing additional theoretical properties that exploit the underlying mathematical structure of such problems. Androulakis *et al.* (1996) developed a distributed version of the GOP, and discussed the key theoreti-

cal and implementation issues along with extensive computational results on large scale indefinite quadratic and pooling/blending problems. Shectman and Sahinidis (1996) proposed a finite algorithm for separable concave programs, discussed the design of such branch and bound approaches, and presented computational results employing domain reduction tests. McKinnon *et al.* (1996) addressed the global optimization in phase and chemical reaction equilibrium using interval analysis coupled with the tangent plane stability criterion of Gibbs. Hua *et al.* (1996) applied an interval analysis method for the phase stability computations of binary and ternary mixtures using activity coefficient models. Mockus and Reklaitis (1996) proposed a continuous formulation for the short term batch scheduling problem, and developed a global optimization approach based on the Bayesian heuristic. Lucia and Xu (1996) studied nonconvexity issues in sparse successive quadratic programming, and Banga and Seider (1996) introduced a stochastic global optimization approach which they applied to the optimal design of a fermentation process, phase equilibrium problems and optimal control problems. Epperly and Swaney (1996a), (1996b) proposed a branch and bound method with a new linear programming underestimating problem, and provided extensive computational studies. This approach is applicable to NLPs in factorable form which includes quadratic objective and constraints, and twice differentiable transcendental functions. Visweswaran and Floudas (1996a) proposed new formulations for the GOP algorithm which are based on a branch and bound framework. These allow for the implicit solution of the relaxed dual problems, which are formulated in a single MILP model, and feature a linear branching scheme. Visweswaran and Floudas (1996b) discussed the implementation issues of the GOP and provide extensive computational experience on a variety of chemical engineering problems. Byrne and Bogle (1996a), (1996b) and Vaidyanathan and El-Halwagi (1996) proposed global optimization methods that are based on interval analysis for constrained NLPs and MINLPs respectively. Liu *et al.* (1996) proposed an approach for planning of chemical process networks which is based on global concave minimization. The approach is based on their earlier work on finite global optimization approaches and their computational studies revealed the efficiency of the proposed approach. Ierapetritou and Pistikopoulos (1996) studied the global optimization of stochastic planning, scheduling and design problems, applied the decomposition-based approach, GOP, and demonstrated that significant reductions in the number of relaxed dual problems can be achieved by exploiting the mathematical structure further. Iyer and Grossmann (1996) extended the global optimization approach of Quesada and Grossmann (1993) to the multiperiod heat exchanger networks that feature fixed configuration, linear cost functions, arithmetic mean driving forces, and isothermal mixing. Quesada and Grossmann (1996) studied further the use of alternative bounding approximations and applied them to a variety of engineering design problems that include structural design, batch processes, layout design, and portfolio problems. Smith and Pantelides (1996) proposed a symbolic manipulation algorithm for the au-

tomatic reformulation of algebraic constraints and introduced a spatial type branch and bound approach within the gPROMS framework.

Maranas and Floudas (1997) proposed a global optimization approach for generalized geometric programming problems that have many applications in robust control and engineering design problems. McDonald and Floudas (1997) proposed the combination of the Gibbs free energy minimization problem with the stability criterion problem, developed a special purpose program GLOPEQ, and performed an extensive computational study on difficult phase equilibrium problems. Harding *et al.* (1997) proposed novel formulations and global optimization methods for the enclosure of all homogeneous azeotropes. This approach offers a theoretical guarantee of enclosing all homogeneous azeotopes and it was applied to a variety of activity coefficient models. Androulakis *et al.* (1997a) proposed a novel global optimization approach based on atomistic level modeling and the αBB principles for the structure prediction of oligopeptides. Androulakis *et al.* (1997b) proposed a decomposition based global optimization approach for the peptide docking problem of amino acids into the binding sites of the HLA-DR1 protein. Harding and Floudas (1997) proposed novel global optimization approaches for the batch design under uncertainty of multiproduct and multipurpose plants. These approaches are based on the exact calculation of the α parameters within the αBB global optimization framework and they result in the tightest possible underestimators. Epperly *et al.* (1997) proposed a global optimization approach for stochastic batch design problems that is based on a reduction of the number of variables needed for branching as suggested in their reduced space branch and bound approach (Epperly and Pistikopoulos (1997)). Sriniwas and Arkun (1997) introduced a global optimization approach based on the GOP principles for the nonlinear model predictive control problems with polynomial ARX models. VanAntwerp *et al.* (1997) introduced a global optimization approach based on branch and bound for the design of a robust controller to time-varying nonlinear plant perturbations. Smith and Pantelides (1997) studied further their reformulation/spatial branch and bound approach for the global optimization of MINLP problems.

Klepeis *et al.* (1998) proposed new global optimization approaches for the structure prediction of solvated peptides using the area and volume accessible to the solvent models. A review of the global optimization activities in the areas of protein folding and peptide docking can be found in Floudas *et al.* (1999b). Esposito and Floudas (1998) proposed the first global optimization method for the parameter estimation and data reconcilliation of nonlinear algebraic models using the principles of the αBB approach. McKinnon and Mongeau (1998) proposed a generic global optimization approach for the phase and chemical reaction equilibrium problem that is based on interval analysis and combines the stability criterion with the minimization of the Gibbs free energy. Hua *et al.* (1998) applied an interval analysis method for the phase stability computations of binary and ternary mixtures using equation of state models. Maier *et al.* (1998) applied an interval analysis based approach for the enclosure of homogeneous azeotropes. They employed the formulations

proposed by Harding *et al.* (1997) and studied systems with activity coefficient and equation of state models. Meyer and Swartz (1998) proposed a new approach for testing convexity for phase equilibrium problems that can be coupled with the global optimization methods of McDonald and Floudas (1994), (1995a), (1995b), (1995c), (1997). Shectman and Sahinidis (1998) proposed a finite global optimization method for separable concave problems. Yamada and Hara (1998) proposed a global optimization approach based on the triangle covering for H-infinity control with constant diagonal scaling. Androulakis and Floudas (1998) studied the parallel computation issues that arise using the αBB global optimization approach. Adjiman *et al.* (1998a) proposed several new rigorous methods for the calculation of the α parameters for (i) uniform diagonal shift of the hessian matrix and (ii) non-uniform diagonal shift of the hessian matrix, and they established their potential trade-offs. Adjiman *et al.* (1998b) presented the detailed implementation of the αBB approach and computational studies in process design problems such as heat exchanger networks, reactor-separator networks, and batch design under uncertainty. Westerlund *et al.* (1998) proposed an extended cutting plane approach for the global optimization of pseudoconvex MINLP problems. Zamora and Grossmann (1998a) proposed a hybrid branch-and-bound and outer approximation method for the global optimization of heat exchanger networks with no stream splits. Zamora and Grossmann (1998b) introduced a deterministic branch and bound approach for structured process systems that have univariate concave, bilinear and linear fractional terms. They proposed several properties of the contraction operation, embedded them in the global optimization algorithm and studied the contraction effects on several applications. Adjiman *et al.* (1998d) proposed two novel global optimization approaches for nonconvex mixed-integer nonlinear programming problems. The first approach is for separable continuous and integer domains while the second approach addresses general mixed-integer nonlinear problems. Klepeis and Floudas (1999a) presented a detailed comparative study of the putative global minimum structures of solvated oligopeptides where the solvation models employed were based on the area and volume accessible to the solvent. Klepeis and Floudas (1999b) proposed a novel deterministic global optimization approach for free energy calculations of peptides and presented extensive computational studies on solvated and unsolvated pentapeptides. Westerberg and Floudas (1999a), (1999b) introduced a global optimization framework for the enclosure of all transition states of potential energy hypersurfaces, and studied the reaction pathways, and the dynamics of folding solvated and unsolvated oligopeptides. Smith and Pantelides (1999) presented a symbolic reformulation/spatial branch and bound approach for the global optimization of nonconvex MINLPs. Adhya *et al.* (1999) introduced a lagrangian approach to address small and medium size pooling problems. Floudas (1999a) and Floudas and Pardalos (1999) reviewed the recent global optimization developments in process design and control. Floudas (1999b) reviewed the advances in enclosing all solutions of nonlinear constrained systems of equations via deterministic global optimization approaches. Klepeis *et al.* (1999) introduced deterministic

global optimization approaches to the NMR structure refinement and studied the structure prediction of Compstatin. Harding and Floudas (1999) developed rigorous global optimization approaches for the phase stability problem when cubic equations of state are used. Harding and Floudas (1999a) introduced deterministic global optimization approaches for enclosing all heterogeneous and reactive azeotropes. Esposito and Floudas (1999) proposed two novel global optimization approaches, based on collocation and integration, for the parameter estimation and optimal control of differential-algebraic models. Floudas *et al.* (1999) introduced a handbook of global optimization test problems that cover a broad spectrum of engineering and applied science applications.

1.9 SCOPE OF THE BOOK

The rest of this book consists of an introduction to the basics of global optimization, presented in Chapter 2, and five main parts. Part I studies global optimization problems of special structure defined as biconvex and bilinear programming problems. Part II focuses on the global optimization of generalized geometric programming problems. Part III addresses general twice continuously differentiable nonlinear optimization problems. Part IV studies global optimization problems that exhibit mixed-integer nonlinear structure. Part V addresses the enclosure of all solutions of constrained nonlinear equations of equations via deterministic global optimization. All five parts contain theoretical advances on deterministic global optimization coupled with extensive computational studies in a variety of applications areas.

2 BASIC CONCEPTS OF GLOBAL OPTIMIZATION

This chapter introduces the reader to the basic concepts that form the foundation for the global optimization theoretical developments, methods and applications. In section 2.1 the fundamentals of convex sets are discussed. In section 2.2 the subject of convex and concave functions is presented. In section 2.3 the definitions and properties of quasiconvex and pseudoconvex functions are discussed. Section 2.4 presents the definition and properties of the convex envelopes. Section 2.5 presents the definitions of the local, global optima and ϵ-global optima. Finally, section 2.6 introduces the reader to the basics of difference of convex functions denoted as D.C. functions.

2.1 CONVEX SETS

This section introduces the fundamental concept of convex sets, and presents their basic properties.

2.1.1 Basic Definitions

Definition 2.1.1 (Line) *Let the vectors $\boldsymbol{x}_1$, $\boldsymbol{x}_2 \in \Re^n$. The line through $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ is defined as the set:*

$$\{\boldsymbol{x} \mid \boldsymbol{x} = (1-\lambda)\,\boldsymbol{x}_1 + \lambda\,\boldsymbol{x}_2,\ \lambda \in \Re\}. \tag{2.1}$$

C. A. Floudas, *Deterministic Global Optimization*
© Springer Science+Business Media Dordrecht 2000

Definition 2.1.2 (Closed Line Segment) *Let the vectors $\boldsymbol{x}_1, \boldsymbol{x}_2 \in \Re^n$. The closed line segment through $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ is defined as the set:*

$$\{\, \boldsymbol{x} \mid \boldsymbol{x} = (1-\lambda)\,\boldsymbol{x}_1 + \lambda\,\boldsymbol{x}_2,\ 0 \le \lambda \le 1 \,\}. \tag{2.2}$$

The *open*, *closed-open*, and *open-closed* line segments can be defined similarly by modifying the inequalities for λ.

Definition 2.1.3 (Halfspace) *Let the vector $\mathbf{c} \in \Re^n$, $\mathbf{c} \neq 0$, and the scalar $z \in \Re$. The open halfspace in $\Re^n$ is defined as the set:*

$$\{\, \boldsymbol{x} \mid \boldsymbol{c}^t\,\boldsymbol{x} < z\,,\ \boldsymbol{x} \in \Re^n \,\}. \tag{2.3}$$

The closed halfspace in $\Re^n$ is defined as the set:

$$\{\, \boldsymbol{x} \mid \boldsymbol{c}^t\,\boldsymbol{x} \le z\,,\ \boldsymbol{x} \in \Re^n \,\}. \tag{2.4}$$

Definition 2.1.4 (Hyperplane) *The hyperplane in $\Re^n$ is defined as the set:*

$$\{\, \boldsymbol{x} \mid \boldsymbol{c}^t\,\boldsymbol{x} = z\ ,\ \boldsymbol{x} \in \Re^n \,\}. \tag{2.5}$$

Definition 2.1.5 (Polytope and Polyhedron) *The intersection of a finite number of closed halfspaces in $\Re^n$ is defined as a polytope. A bounded polytope is called a polyhedron.*

Definition 2.1.6 (Convex Set) *A set $S \in \Re^n$ is said to be convex if the closed line segment joining any two points $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ of the set S, that is, (1 - λ) $\boldsymbol{x}_1$ + λ $\boldsymbol{x}_2$, belongs to the set S for each λ such that $0 \le \lambda \le 1$.*

Illustration 2.1.1 The following are some examples of convex sets:

(i) Line

(ii) Open and closed half-space

(iii) Polytope, polyhedron

(iv) All points inside or on the circle

(v) All points inside or on a polygon

Lemma 2.1.1 (Properties of Convex Sets) *Let S_1 and S_2 be convex sets in $\Re^n$. Then,*

(i) *The intersection $S_1 \cap S_2$ is a convex set.*

(ii) *The sum $S_1 + S_2$ of two convex sets is a convex set.*

(iii) *The product $\theta\, S_1$ of the real number θ and the set S_1 is a convex set.*

(iv) *the difference $S_1 - S_2$ of two convex sets is a convex set.*

2.1.2 Convex Combination and Convex Hull

Definition 2.1.7 (Convex Combination) *Let $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_r\}$ be any finite set of points in $\Re^n$. A convex combination of this set is a point of the form:*

$$\lambda_1 \boldsymbol{x}_1 + \ldots + \lambda_r \boldsymbol{x}_r \tag{2.6}$$

$$\lambda_1 + \ldots + \lambda_r = 1 \tag{2.7}$$

$$\lambda_1, \ldots, \lambda_r \geq 0. \tag{2.8}$$

Definition 2.1.8 (Convex Hull) *Let S be a set (convex or nonconvex) in $\Re^n$. The convex hull, $H(S)$, of S is defined as the intersection of all convex sets in $\Re^n$ which contain S as a subset.*

Theorem 2.1.1 *The convex hull, $H(S)$, of S is defined as the set of all convex combinations of S. Then $\boldsymbol{x} \in H(S)$ if and only if $\boldsymbol{x}$ can be represented as:*

$$\boldsymbol{x} = \sum_{i=1}^{r} \lambda_i \boldsymbol{x}_i \tag{2.9}$$

$$\sum_{i=1}^{r} \lambda_i = 1 \tag{2.10}$$

$$\lambda_i \geq 0, \, i = 1, \ldots, r \tag{2.11}$$

$$\boldsymbol{x}_i \in S, \, i = 1, \ldots, r \tag{2.12}$$

where r is a positive integer.

Remark 2.1.1 Any point $\boldsymbol{x}$ in the convex hull of a set S in $\Re^n$ can be written as a convex combination of at most $n+1$ points in S as demonstrated by the following theorem.

Theorem 2.1.2 (Caratheodory) *Let S be a set (convex or nonconvex) in $\Re^n$. If $\boldsymbol{x} \in H(S)$, then it can be expressed as:*

$$\boldsymbol{x} = \sum_{i=1}^{n+1} \lambda_i \boldsymbol{x}_i \tag{2.13}$$

$$\sum_{i=1}^{n+1} \lambda_i = 1 \tag{2.14}$$

$$\lambda_i \geq 0, \, i = 1, \ldots, n+1 \tag{2.15}$$

$$\boldsymbol{x}_i \in S, \, i = 1, \ldots, n+1. \tag{2.16}$$

2.2 CONVEX AND CONCAVE FUNCTIONS

This section presents (i) the definitions and properties of convex and concave functions, (ii) the definitions of continuity, semi-continuity and subgradients, and (iii) the definitions and properties of differentiable convex and concave functions.

2.2.1 Basic Definitions

Definition 2.2.1 (Convex Function) *Let S be a convex subset of $\Re^n$, and $f(\boldsymbol{x})$ be a real valued function defined on S. The function $f(\boldsymbol{x})$ is said to be convex if for any $\boldsymbol{x}_1, \boldsymbol{x}_2 \in S$, and $0 \leq \lambda \leq 1$, we have*

$$f((1-\lambda)\boldsymbol{x}_1 + \lambda \boldsymbol{x}_2) \leq (1-\lambda)f(\boldsymbol{x}_1) + \lambda f(\boldsymbol{x}_2). \tag{2.17}$$

This inequality is called *Jensen's inequality* after the Danish mathematician who first introduced it.

Definition 2.2.2 (Strictly Convex Function) *Let S be a convex subset of $\Re^n$, and $f(\boldsymbol{x})$ be a real valued function defined on S. The function $f(\boldsymbol{x})$ is said to be strictly convex if for any $\boldsymbol{x}_1, \boldsymbol{x}_2 \in S$, and $0 < \lambda < 1$, we have*

$$f((1-\lambda)\boldsymbol{x}_1 + \lambda \boldsymbol{x}_2) < (1-\lambda)f(\boldsymbol{x}_1) + \lambda f(\boldsymbol{x}_2). \tag{2.18}$$

Remark 2.2.1 A strictly convex function on a subset S of $\Re^n$ is convex on S. The converse however is not true. For instance, a linear function is convex but not strictly convex.

Definition 2.2.3 (Concave Function) *Let S be a convex subset of $\Re^n$, and $f(\boldsymbol{x})$ be a real valued function defined on S. The function $f(\boldsymbol{x})$ is said to be concave if for any $\boldsymbol{x}_1, \boldsymbol{x}_2 \in S$, and $0 \leq \lambda \leq 1$, we have*

$$f((1-\lambda)\boldsymbol{x}_1 + \lambda \boldsymbol{x}_2) \geq (1-\lambda)f(\boldsymbol{x}_1) + \lambda f(\boldsymbol{x}_2). \tag{2.19}$$

Remark 2.2.2 The function $f(\boldsymbol{x})$ is concave on S if and only if $-f(\boldsymbol{x})$ is convex on S. Then, the results obtained for convex functions can be modified into results for concave functions by multiplication by -1 and vice versa.

Definition 2.2.4 (Strictly Concave Function) *Let S be a convex subset of $\Re^n$, and $f(\boldsymbol{x})$ be a real valued function defined on S. The function $f(\boldsymbol{x})$ is said to be strictly concave if for any $\boldsymbol{x}_1, \boldsymbol{x}_2 \in S$, and $0 < \lambda < 1$, we have*

$$f((1-\lambda)\boldsymbol{x}_1 + \lambda \boldsymbol{x}_2) > (1-\lambda)f(\boldsymbol{x}_1) + \lambda f(\boldsymbol{x}_2). \tag{2.20}$$

Definition 2.2.5 (Nonconvex Function) *Let S be a convex subset of $\Re^n$, and $f(\boldsymbol{x})$ be a real valued function defined on S. The function $f(\boldsymbol{x})$ is said to be nonconvex if it is not convex.*

Remark 2.2.3 A general nonconvex function contains convex and concave parts and as a result it exhibits multiple minima and maxima. A concave function defined within box constraints on all variables is a special case of a nonconvex function since it has a single maximum but several minima that correspond to the vertex points of the feasible region.

2.2.2 *Properties of Convex and Concave Functions*

Convex functions can be combined in a number of ways to produce new convex functions as illustrated by the following:

(i) Let $f_1(\boldsymbol{x}), \ldots, f_n(\boldsymbol{x})$ be convex functions on a convex subset S of $\Re^n$. Then, their summation

$$f_1(\boldsymbol{x}) + \ldots + f_n(\boldsymbol{x}) \tag{2.21}$$

is convex. Furthermore, if at least one $f_i(\boldsymbol{x})$ is strictly convex on S, then their summation is strictly convex.

(ii) Let $f(\boldsymbol{x})$ be convex (strictly convex) on a convex subset S of $\Re^n$, and λ is a positive number. Then, $\lambda\, f(\boldsymbol{x})$ is convex (strictly convex).

(iii) Let $f(\boldsymbol{x})$ be convex (strictly convex) on a convex subset S of $\Re^n$, and $g(y)$ be an increasing convex function defined on the range of $f(\boldsymbol{x})$ in $\Re$. Then, the composite function $g(f(\boldsymbol{x}))$ is convex (strictly convex) on S.

(iv) Let $f_1(\boldsymbol{x}), \ldots, f_n(\boldsymbol{x})$ be convex functions and bounded from above on a convex subset S of $\Re^n$. Then, the pointwise supremum function

$$f(\boldsymbol{x}) = \max\{f_1(\boldsymbol{x}), \ldots, f_n(\boldsymbol{x})\} \tag{2.22}$$

is a convex function on S.

(v) Let $f_1(\boldsymbol{x}), \ldots, f_n(\boldsymbol{x})$ be concave functions and bounded from below on a convex subset S of $\Re^n$. Then, the pointwise infimum function

$$f(\boldsymbol{x}) = \min\{f_1(\boldsymbol{x}), \ldots, f_n(\boldsymbol{x})\} \tag{2.23}$$

is a concave function on S.

(vi) Let S_1 be a convex set in $\Re^n$, S_2 be a convex set in $\Re^m$, and $f(\boldsymbol{x}, \boldsymbol{y})$ be a real valued function on $S_1 \times S_2$. Then, the function

$$\phi(\boldsymbol{x}) = \min_{\boldsymbol{y}} f(\boldsymbol{x}, \boldsymbol{y}) \tag{2.24}$$

is convex on S_1.

(vii) Let $f_1(\boldsymbol{x}), \ldots, f_n(\boldsymbol{x})$ be concave functions on a convex subset S of $\Re^n$. Then, their geometric mean function

$$f(\boldsymbol{x}) = (\prod_i f_i(\boldsymbol{x}))^{1/n} \tag{2.25}$$

is a concave function on S.

Definition 2.2.6 (Epigraph of a function) *Let S be a non-empty set in $\Re^n$. The epigraph of a function $f(\boldsymbol{x})$, denoted by epi(f), is a subset of $\Re^{n+1}$ defined as the set of $(n+1)$ vectors $(\boldsymbol{x}, \boldsymbol{y})$:*

$$\{ (\boldsymbol{x}, \boldsymbol{y}) : f(\boldsymbol{x}) \leq \boldsymbol{y}, \quad \boldsymbol{x} \in S, \quad \boldsymbol{y} \in \Re \}. \tag{2.26}$$

Definition 2.2.7 (Hypograph of a function) *The hypograph of $f(\boldsymbol{x})$, denoted by hyp(f), is a subset of $\Re^{n+1}$ defined as the set of $(n+1)$ vectors $(\boldsymbol{x}, \boldsymbol{y})$:*

$$\{ (\boldsymbol{x}, \boldsymbol{y}) : f(\boldsymbol{x}) \geq \boldsymbol{y}, \quad \boldsymbol{x} \in S, \quad \boldsymbol{y} \in \Re \}. \tag{2.27}$$

Theorem 2.2.1 *Let S be a non-empty set in $\Re^n$. The function $f(\boldsymbol{x})$ is convex if and only if epi(f) is a convex set.*

Remark 2.2.4 The epigraph of a convex function and the hypograph of a concave function are convex sets.

2.2.3 Continuity and Semi-continuity

Definition 2.2.8 (Continuous function) *Let S be a subset of $\Re^n$, $\boldsymbol{x}^0 \in S$ and $f(\boldsymbol{x})$ a real valued function defined on S. $f(\boldsymbol{x})$ is continuous at $\boldsymbol{x}^0$ if either of the following equivalent conditions hold:*

Condition 1: For each $\epsilon_1 > 0$, there exists an $\epsilon_2 > 0$:
$\| \boldsymbol{x} - \boldsymbol{x}^0 \| < \epsilon_2$, $\boldsymbol{x} \in S$ *implies that*

$$-\epsilon_1 < f(\boldsymbol{x}) - f(\boldsymbol{x}^0) < \epsilon_1 . \tag{2.28}$$

Condition 2: For each sequence $\boldsymbol{x}^1, \boldsymbol{x}^2, \ldots \boldsymbol{x}^n$ $(\boldsymbol{x} \in S)$ converging to $\boldsymbol{x}^0$,

$$\lim_{n \to \infty} f(\boldsymbol{x}^n) = f(\lim_{n \to \infty} \boldsymbol{x}^n) = f(\boldsymbol{x}^0) , \tag{2.29}$$

$f(\boldsymbol{x})$ is continuous on S if it is continuous at each $\boldsymbol{x}^0 \in S$.

Definition 2.2.9 (Lower semi-continuous function) *$f(\boldsymbol{x})$ is lower semi-continuous at $\boldsymbol{x}^0$ if either of the following equivalent conditions hold:*

Condition 1: For each $\epsilon_1 > 0$, there exists an $\epsilon_2 > 0$:
$\| \boldsymbol{x} - \boldsymbol{x}^0 \| < \epsilon_2$, $\boldsymbol{x} \in S$ *implies that*

$$-\epsilon_1 < f(\boldsymbol{x}) - f(\boldsymbol{x}^0) . \tag{2.30}$$

Condition 2: For each sequence $\boldsymbol{x}^1, \boldsymbol{x}^2, \ldots, \boldsymbol{x}^n$ $(\boldsymbol{x} \in S)$ converging to $\boldsymbol{x}^0$,

$$\lim_{n \to \infty} \inf f(\boldsymbol{x}^n) \geq f(\lim_{n \to \infty} \boldsymbol{x}^n) = f(\boldsymbol{x}^0) \tag{2.31}$$

where $\lim_{n \to \infty} \inf f(\boldsymbol{x}^n)$ is the infimum of the limit points of the sequence $f(\boldsymbol{x}^1), f(\boldsymbol{x}^2), \ldots, f(\boldsymbol{x}^n)$.

$f(\boldsymbol{x})$ is lower semi-continuous on S if it is lower semi-continuous at each $\boldsymbol{x}^0 \in S$.

Definition 2.2.10 (Upper semi-continuous function) *$f(\boldsymbol{x})$ is upper semi-continuous at $\boldsymbol{x}^0$ if either of the following equivalent conditions hold:*

Condition 1: For each $\epsilon_1 > 0$, there exists an $\epsilon_2 > 0$:
$\| \boldsymbol{x} - \boldsymbol{x}^0 \| < \epsilon_2$, $\boldsymbol{x} \in S$ implies that

$$f(\boldsymbol{x}) - f(\boldsymbol{x}^0) < \epsilon_1. \tag{2.32}$$

Condition 2: For each sequence $\boldsymbol{x}^1, \boldsymbol{x}^2, \ldots, \boldsymbol{x}^n$ ($\boldsymbol{x} \in S$) converging to $\boldsymbol{x}^0$,

$$\lim_{n\to\infty} \sup f(\boldsymbol{x}^n) \leq f(\lim_{n\to\infty} \boldsymbol{x}^n) = f(\boldsymbol{x}^0) \tag{2.33}$$

where $\lim_{n\to\infty} \sup f(\boldsymbol{x}^n)$ is the supremum of the limit points of the sequence $f(\boldsymbol{x}^1), f(\boldsymbol{x}^2), \ldots, f(\boldsymbol{x}^n)$.

$f(\boldsymbol{x})$ is upper semi-continuous on S if it is upper semi-continuous at each $\boldsymbol{x}^0 \in S$.

Theorem 2.2.2 *Let S be a non-empty convex set in $\Re^n$ and $f(\boldsymbol{x})$ be a convex function. Then, $f(\boldsymbol{x})$ is continuous on the interior of S.*

Remark 2.2.5 Convex and concave functions may not be continuous everywhere but the points of discontinuity have to be on the boundary of S.

Theorem 2.2.3 *Let $f_i(\boldsymbol{x})$ be a family of lower (upper) semi-continuous functions on S. Then*

(i) *Its least upper bound (greatest lower bound)*

$$f(\boldsymbol{x}) = \sup_i f_i(\boldsymbol{x}) \quad (f'(\boldsymbol{x}) = \inf_i f_i(\boldsymbol{x})) \tag{2.34}$$

is lower (upper) semi-continuous on S.

(ii) *If the family $f_i(\boldsymbol{x})$ is finite, its greatest lower bound (least upper bound)*

$$f'(\boldsymbol{x}) = \inf_i f_i(\boldsymbol{x}) \quad (f(\boldsymbol{x}) = \sup_i f_i(\boldsymbol{x})) \tag{2.35}$$

is lower (upper) semi-continuous on S.

2.2.4 Differentiable Convex and Concave Functions

Definition 2.2.11 (Differentiable Function) *Let S be a non-empty open set in $\Re^n$, $f(\boldsymbol{x})$ be a function defined on S, and $\boldsymbol{x}^0 \in S$. Then, $f(\boldsymbol{x})$ is differentiable at $\boldsymbol{x}^0$ if for all $(\Delta\boldsymbol{x} = \boldsymbol{x} - \boldsymbol{x}^0) \in \Re^n$ such that $(\boldsymbol{x}^0 + \Delta\boldsymbol{x}) \in S$ we have*

$$f(\boldsymbol{x}^0 + \Delta\boldsymbol{x}) = f(\boldsymbol{x}^0) + \nabla f(\boldsymbol{x}^0)\Delta\boldsymbol{x} + \alpha(\boldsymbol{x}^0, \Delta\boldsymbol{x}) \| \Delta\boldsymbol{x} \| \tag{2.36}$$

with

$$\lim_{\Delta \boldsymbol{x} \to 0} \alpha(\boldsymbol{x}^0, \Delta \boldsymbol{x}) = 0 \tag{2.37}$$

where $\alpha(\boldsymbol{x}^0, \Delta \boldsymbol{x})$ *is a function of* $\Delta \boldsymbol{x}$, *and* $\nabla f(\boldsymbol{x}^0)$ *is the gradient of* $f(\boldsymbol{x})$ *evaluated at* $\boldsymbol{x}^0$ *(the* n*-dimensional vector of the partial derivatives of* $f(\boldsymbol{x})$ *with respect to* $x_1, x_2, \ldots, x_n$ *); that is,*

$$\nabla f(\boldsymbol{x}^0) = \left(\frac{\partial f(\boldsymbol{x}^0)}{\partial x_1}, \ldots, \frac{\partial f(\boldsymbol{x}^0)}{\partial x_n} \right). \tag{2.38}$$

$f(\boldsymbol{x})$ *is differentiable on* S *if it is differentiable at each* $\boldsymbol{x}^0 \in S$.

Remark 2.2.6 If $f(\boldsymbol{x})$ is differentiable at $\boldsymbol{x}^0$, then $f(\boldsymbol{x})$ is continuous at $\boldsymbol{x}^0$ and $\nabla f(\boldsymbol{x}^0)$ exists. However, the converse is not true.

Remark 2.2.7 If $\nabla f(\boldsymbol{x}^0)$ exists and $\nabla f(\boldsymbol{x})$ is continuous at $\boldsymbol{x}^0$, then $f(\boldsymbol{x})$ is differentiable at $\boldsymbol{x}^0$.

Definition 2.2.12 (Twice differentiable function) *Let* S *be a non-empty open set in* $\Re^n$, $f(\boldsymbol{x})$ *be a function defined on* S, *and* $\boldsymbol{x}^0 \in S$. $f(\boldsymbol{x})$ *is twice differentiable at* $\boldsymbol{x}^0$ *if for all* $(\Delta \boldsymbol{x} = \boldsymbol{x} - \boldsymbol{x}^0) \in \Re^n$ *such that* $(\boldsymbol{x}^0 + \Delta \boldsymbol{x}) \in S$ *we have*

$$\begin{aligned} f(\boldsymbol{x}^0 + \Delta \boldsymbol{x}) = f(\boldsymbol{x}^0) \quad &+ \quad \nabla f(\boldsymbol{x}^0) \Delta \boldsymbol{x} \\ &+ \quad \frac{\Delta \boldsymbol{x}^T \nabla^2 f(\boldsymbol{x}^0) \Delta \boldsymbol{x}}{2} + \beta(\boldsymbol{x}^0, \Delta \boldsymbol{x}) \| \Delta \boldsymbol{x} \|^2 \end{aligned} \tag{2.39}$$

with

$$\lim_{\Delta \boldsymbol{x} \to 0} \beta(\boldsymbol{x}^0, \Delta \boldsymbol{x}) = 0 \tag{2.40}$$

where $\beta(\boldsymbol{x}^0, \Delta \boldsymbol{x})$ *is a function of* $\Delta \boldsymbol{x}$, *and* $\nabla^2 f(\boldsymbol{x}^0)$ *is the Hessian of* $f(\boldsymbol{x})$ *evaluated at* $\boldsymbol{x}^0$, *that is, an* $n \times n$ *matrix whose* ij^{th} *element is* $\frac{\partial^2 f(\boldsymbol{x}^0)}{\partial x_i \partial x_j}$.

Remark 2.2.8 If $\nabla f(\boldsymbol{x})$ is differentiable at $\boldsymbol{x}^0$ (i.e. it has continuous partial derivatives at $\boldsymbol{x}^0$), then $f(\boldsymbol{x})$ is twice differentiable at $\boldsymbol{x}^0$. If $\nabla^2 f(\boldsymbol{x})$ is continuous at $\boldsymbol{x}^0$, then

$$\frac{\partial^2 f(\boldsymbol{x}^0)}{\partial x_i \partial x_j} = \frac{\partial}{\partial x_i} \left(\frac{\partial f(\boldsymbol{x}^0)}{\partial x_j} \right), \quad \text{and} \tag{2.41}$$

$\nabla^2 f(\boldsymbol{x}^0)$ is symmetric $((\nabla^2 f(\boldsymbol{x}))_{ij} = (\nabla^2 f(\boldsymbol{x}))_{ji})$.

Theorem 2.2.4 *Let* S *be a non-empty open set in* $\Re^n$, *and* $f(\boldsymbol{x})$ *a differentiable function at* $\boldsymbol{x}^0 \in S$.

(i) *If* $f(\boldsymbol{x})$ *is convex at* $\boldsymbol{x}^0 \in S$, *then*

$$f(\boldsymbol{x}) - f(\boldsymbol{x}^0) \geq \nabla f(\boldsymbol{x}^0)(\boldsymbol{x} - \boldsymbol{x}^0) \quad \forall\, x \in S. \tag{2.42}$$

(ii) *If $f(\boldsymbol{x})$ is concave at $\boldsymbol{x}^0 \in S$, then*

$$f(\boldsymbol{x}) - f(\boldsymbol{x}^0) \leq \nabla f(\boldsymbol{x}^0)(\boldsymbol{x} - \boldsymbol{x}^0) \quad \forall\, x \in S. \tag{2.43}$$

Theorem 2.2.5 *Let S be a non-empty open set in $\Re^n$, and $f(\boldsymbol{x})$ a differentiable function on S.*

(i) *$f(\boldsymbol{x})$ is convex on S if and only if*

$$f(\boldsymbol{x}^2) - f(\boldsymbol{x}^1) \geq \nabla f(\boldsymbol{x}^1)(\boldsymbol{x}^2 - \boldsymbol{x}^1) \quad \forall\, \boldsymbol{x}^1, \boldsymbol{x}^2 \in S. \tag{2.44}$$

(ii) *$f(\boldsymbol{x})$ is concave on S if and only if*

$$f(\boldsymbol{x}^2) - f(\boldsymbol{x}^1) \leq \nabla f(\boldsymbol{x}^1)(\boldsymbol{x}^2 - \boldsymbol{x}^1) \quad \forall\, \boldsymbol{x}^1, \boldsymbol{x}^2 \in S. \tag{2.45}$$

Remark 2.2.9 The above two theorems can be directly extended to strictly convex and strictly concave functions by replacing the inequalities $\geq$ and $\leq$ with strict inequalities $>$ and $<$.

Theorem 2.2.6 *Let S be a non-empty open set in $\Re^n$, and $f(\boldsymbol{x})$ be a twice differentiable function at $\boldsymbol{x}^0 \in S$.*

(i) *If $f(\boldsymbol{x})$ is convex at $\boldsymbol{x}^0$, then*

$$\nabla^2 f(\boldsymbol{x}^0) \text{ is positive semi-definite (i.e., } z^T \nabla^2 f(\boldsymbol{x}^0) \cdot z \geq 0,\ \forall\, z \in \Re^n) \tag{2.46}$$

(ii) *If $f(\boldsymbol{x})$ is concave at $\boldsymbol{x}^0$, then*

$$\nabla^2 f(\boldsymbol{x}^0) \text{ is negative semi-definite (i.e. } z^T \nabla^2 f(\boldsymbol{x}^0) \cdot z \leq 0,\ \forall\, z \in \Re^n) \tag{2.47}$$

Theorem 2.2.7 *Let S be a non-empty open set in $\Re^n$, and $f(\boldsymbol{x})$ be a twice differentiable function at $\boldsymbol{x}^0 \in S$.*

(i) *$f(\boldsymbol{x})$ is convex on S if and only if*

$$\nabla^2 f(\boldsymbol{x}) \quad \text{is positive semi-definite on } S \text{ for all } \boldsymbol{x} \in S. \tag{2.48}$$

(ii) *$f(\boldsymbol{x})$ is concave on S if and only if*

$$\nabla^2 f(\boldsymbol{x}) \quad \text{is negative semi-definite on } S \text{ for all } \boldsymbol{x} \in S. \tag{2.49}$$

Remark 2.2.10 **(i)** If $f(\boldsymbol{x})$ is strictly convex at $\boldsymbol{x}^0$, then $\nabla^2 f(\boldsymbol{x}^0)$ is positive semi-definite (not necessarily positive definite).

(ii) If $\nabla^2 f(\boldsymbol{x}^0)$ is positive definite, then $f(\boldsymbol{x})$ is strictly convex at $\boldsymbol{x}^0$.

(iii) If $f(\boldsymbol{x})$ is strictly concave at $\boldsymbol{x}^0$, then $\nabla^2 f(\boldsymbol{x}^0)$ is negative semi-definite (not necessarily negative definite).

(iv) If $\nabla^2 f(\boldsymbol{x}^0)$ is negative definite, then $f(\boldsymbol{x})$ is strictly concave at $\boldsymbol{x}^0$.

Remark 2.2.11 Theorem 2.2.7 provides the conditions for checking the convexity or concavity of a function $f(\boldsymbol{x})$. These conditions correspond to positive semi-definite (PSD) or negative semi-definite (NSD) Hessian of $f(\boldsymbol{x})$ for all $\boldsymbol{x} \in S$ respectively. One test of PSD or NSD Hessian of $f(\boldsymbol{x})$ is based on the sign of eigenvalues of the Hessian. If all eigenvalues are greater than or equal to zero for all $\boldsymbol{x} \in S$, then the Hessian is PSD and hence the function $f(\boldsymbol{x})$ is convex. If all eigenvalues are less or equal than zero for all $\boldsymbol{x} \in S$ then the Hessian is NSD and therefore the function $f(\boldsymbol{x})$ is concave.

2.3 QUASI-CONVEX AND PSEUDO-CONVEX FUNCTIONS

This section presents the definitions, properties and relationships of quasi-convex and pseudo-convex functions.

2.3.1 Quasi-Convex and Quasi-Concave Functions

Let S be a non-empty convex set in $\Re^n$.

Definition 2.3.1 (Quasi-Convex Function) *$f(\boldsymbol{x})$ is quasi-convex if*

$$f[(1-\lambda)\boldsymbol{x}_1 + \lambda \boldsymbol{x}_2] \leq \max[\, f(\boldsymbol{x}_1), f(\boldsymbol{x}_2)\,]$$

$$\text{for all } \lambda \in [0,1] \quad \text{and all } \boldsymbol{x}_1, \boldsymbol{x}_2 \in S.$$

Definition 2.3.2 (Quasi-Concave Function) *$f(\boldsymbol{x})$ is quasi-concave if*

$$f[(1-\lambda)\boldsymbol{x}_1 + \lambda \boldsymbol{x}_2] \geq \min[\, f(\boldsymbol{x}_1), f(\boldsymbol{x}_2)\,]$$

$$\text{for all } \lambda \in [0,1] \quad \text{and all } \boldsymbol{x}_1, \boldsymbol{x}_2 \in S.$$

Note that $f(\boldsymbol{x})$ is quasi-concave if $-f(\boldsymbol{x})$ is quasi-convex.

Remark 2.3.1 A convex function is also quasi-convex since

$$f[(1-\lambda)\boldsymbol{x}_1 + \lambda \boldsymbol{x}_2] \leq (1-\lambda) f(\boldsymbol{x}_1) + \lambda f(\boldsymbol{x}_2) \leq \max[f(\boldsymbol{x}_1), f(\boldsymbol{x}_2)].$$

Similarly a concave function is also quasi-concave since

$$f[(1-\lambda)\boldsymbol{x}_1 + \lambda \boldsymbol{x}_2] \geq (1-\lambda) f(\boldsymbol{x}_1) + \lambda f(\boldsymbol{x}_2) \geq \min[f(\boldsymbol{x}_1), f(\boldsymbol{x}_2)].$$

Theorem 2.3.1 *Consider the function $f(\boldsymbol{x})$ on a convex set $S \in \Re^n$, and*

$$\begin{aligned} S_\alpha &= \{\boldsymbol{x} \mid \boldsymbol{x} \in S,\, f(\boldsymbol{x}) \leq \alpha\}, \quad \text{and} \\ S_\beta &= \{\boldsymbol{x} \mid \boldsymbol{x} \in S,\, f(\boldsymbol{x}) \geq \beta\}. \end{aligned}$$

Then,

(i) *$f(\boldsymbol{x})$ is quasi-convex on S if and only if S_α is convex for each $\alpha \in \Re$.*

(ii) *$f(\boldsymbol{x})$ is quasi-concave on S if and only if S_β is convex for each $\beta \in \Re$.*

Definition 2.3.3 (Strictly quasi-convex function) *$f(\boldsymbol{x})$ is strictly quasi-convex if*

$$f[(1-\lambda)\boldsymbol{x}_1 + \lambda \boldsymbol{x}_2] < \max[f(\boldsymbol{x}_1), f(\boldsymbol{x}_2)]$$

for all $\lambda \in (0,1)$ and all $\boldsymbol{x}_1, \boldsymbol{x}_2 \in S, f(\boldsymbol{x}_1) \neq f(\boldsymbol{x}_2)$.

Definition 2.3.4 (Strictly quasi-concave function) *$f(\boldsymbol{x})$ is strictly quasi-concave if*

$$f[(1-\lambda)\boldsymbol{x}_1 + \lambda \boldsymbol{x}_2] > \min[f(\boldsymbol{x}_1), f(\boldsymbol{x}_2)]$$

for all $\lambda \in (0,1)$ and all $\boldsymbol{x}_1, \boldsymbol{x}_2 \in S, f(\boldsymbol{x}_1) \neq f(\boldsymbol{x}_2)$. Note that $f(\boldsymbol{x})$ is strictly quasi-concave if $-f(\boldsymbol{x})$ is strictly quasi-convex.

Theorem 2.3.2 *Let $f(\boldsymbol{x})$ be a lower (upper) semi-continuous function on the convex set S in $\Re^n$. If $f(\boldsymbol{x})$ is strictly quasi-convex (strictly quasi-concave) on S, then*
$f(\boldsymbol{x})$ is quasi-convex (quasi-concave) on S, but the converse is not true.

Theorem 2.3.3 *Let $f(\boldsymbol{x})$ be a function on the convex set S in $\Re^n$, and let $\boldsymbol{x}^\star \in S$ be a local minimum (maximum) of $f(\boldsymbol{x})$. If $f(\boldsymbol{x})$ is strictly quasi-convex (strictly quasi-concave) on S, then*
$f(\boldsymbol{x}^\star)$ is a global minimum (maximum) of $f(\boldsymbol{x})$ on S.

2.3.2 Properties of Quasi-Convex and Quasi-Concave Functions

Quasi-convex and quasi-concave functions satisfy the following properties:

(i) Let $f(\boldsymbol{x})$ be a quasi-convex (quasi-concave) function on a subset S of $\Re^n$ and $g(y)$ be a nondecreasing function defined on the range of $f(\boldsymbol{x})$ in $\Re$. Then the composite function $g(f(\boldsymbol{x}))$ is quasi-convex (quasi-concave) on S.

(ii) If $f(\boldsymbol{x})$ is either a positive or negative quasi-concave function on a subset S of $\Re^n$, then $\frac{1}{f(\boldsymbol{x})}$ is quasi-convex on S.

(iii) If $f(\boldsymbol{x})$ is either a positive or negative quasi-convex function on a subset S of $\Re^n$, then $\frac{1}{f(\boldsymbol{x})}$ is quasi-concave on S.

Remark 2.3.2 Note that the summation of quasi-convex functions is not necessarily a quasi-convex function as in convex functions. Also, note that the summation of convex and quasi-convex functions is not necessarily a convex or quasi-convex function.

Remark 2.3.3 Convex and concave functions do not satisfy properties (ii) and (iii) of the quasi-convex and quasi-concave functions. For instance, it is true that the reciprocal of a positive concave function is convex, but the reverse does not hold. As an example consider the function $f(\boldsymbol{x}) = e^x$ which is convex and whose reciprocal is also convex.

2.3.3 Differentiable Quasi-Convex, Quasi-Concave Functions

Theorem 2.3.4 *Let $f(\boldsymbol{x})$ be differentiable on a nonempty open convex set S in $\Re^n$. Then, $f(\boldsymbol{x})$ is quasi-convex if and only if for every $\boldsymbol{x}_1, \boldsymbol{x}_2 \in S$*

$$f(\boldsymbol{x}_1) \leq f(\boldsymbol{x}_2) \text{ implies that } (\boldsymbol{x}_1 - \boldsymbol{x}_2)^T \nabla f(\boldsymbol{x}_2) \leq 0.$$

Similarly, f(x) is quasi-concave if and only if for every $\boldsymbol{x}_1, \boldsymbol{x}_2 \in S$

$$f(\boldsymbol{x}_1) \geq f(\boldsymbol{x}_2) \text{ implies that } (\boldsymbol{x}_1 - \boldsymbol{x}_2)^T \nabla f(\boldsymbol{x}_2) \geq 0.$$

For twice differentiable quasi-concave functions $f(\boldsymbol{x})$ on the open, nonempty convex set S in $\Re^n$, a direction $\boldsymbol{z}$ orthogonal to ∇f exhibits the following interesting properties:

(i) If $\overline{\boldsymbol{x}} \in S, \boldsymbol{z} \in \Re^n$ and $\boldsymbol{z}^T \nabla f(\overline{\boldsymbol{x}}) = 0$, then

$$\boldsymbol{z}^T \nabla^2 f(\overline{\boldsymbol{x}}) \boldsymbol{z} \leq 0.$$

(ii) The Hessian of $f(\boldsymbol{x})$ has at most one positive eigenvalue at every $\boldsymbol{x} \in S$.

Remark 2.3.4 From property (ii) we observe that the generalization of concavity to quasi-concavity is equivalent to allowing the existence of at most one positive eigenvalue of the Hessian.

2.3.4 Pseudo-Convex and Pseudo-Concave Functions

Let S be a non-empty open set in $\Re^n$ and let $f(\boldsymbol{x})$ be a differentiable function on S.

Definition 2.3.5 (Pseudo-Convex Function) *$f(\boldsymbol{x})$ is pseudo-convex if for every $\boldsymbol{x}_1, \boldsymbol{x}_2 \in S$,*

$$f(\boldsymbol{x}_1) < f(\boldsymbol{x}_2)$$

implies that $(\boldsymbol{x}_1 - \boldsymbol{x}_2)^T \nabla f(\boldsymbol{x}_2) < 0$.

Definition 2.3.6 (Pseudo-Concave Function) *$f(\boldsymbol{x})$ is pseudo-concave if for every $\boldsymbol{x}_1, \boldsymbol{x}_2 \in S$,*

$$f(\boldsymbol{x}_1) > f(\boldsymbol{x}_2)$$

implies that $(\boldsymbol{x}_1 - \boldsymbol{x}_2)^T \nabla f(\boldsymbol{x}_2) > 0$.

Note that $f(\boldsymbol{x})$ is pseudo-concave if $-f(\boldsymbol{x})$ is pseudo-convex.

2.3.5 Properties of Pseudo-Convex and Pseudo-Concave Functions

Pseudo-convex and pseudo-concave functions exhibit the following properties:

(i) Let $f(\boldsymbol{x})$ be a pseudo-convex (pseudo-concave) function on a subset S of $\Re^n$ and $g(y)$ be a differentiable function defined on the range of $f(\boldsymbol{x})$ in

$\Re$ and which satisfies $g'(y) > 0$. Then the composite function $g(f(x))$ is pseudo-convex (pseudo-concave) on S.

(ii) If $f(x)$ is a positive or negative pseudo-concave function on a subset S of $\Re^n$, then $\frac{1}{f(x)}$ is pseudo-convex on S.

2.3.6 Relationships among Convex, Quasi-Convex and Pseudo-Convex Functions

The relationships among convex, quasi-convex and pseudo-convex functions are summarized in the following:

(i) A convex differentiable function is pseudo-convex,

(ii) A convex function is strictly quasi-convex,

(iii) A convex function is quasi-convex,

(iv) A pseudo-convex function is strictly quasi-convex, and

(v) A strictly quasi-convex function which is lower semi-continuous is quasi-convex.

2.4 CONVEX ENVELOPES

Convex envelopes of functions are of primary importance in global optimization since they represent the uniformly best convex underestimators. This section presents the definition, properties, and key theoretical results on convex envelopes.

Definition 2.4.1 (Convex Envelope) *Let $f(x)$ be a lower semicontinuous function defined on the nonempty convex set $S \in \Re^n$. Then, the convex envelope of $f(x)$ on $S \subset \Re^n$ is a function $\phi(x)$ that satisfies:*

(i) $\phi(x)$ is convex on S.

(ii) $\phi(x) \leq f(x)$ for all $x \in S$.

(iii) If $h(x)$ is any convex function defined on S such that $h(x) \leq f(x)$ for all $x \in S$, then $h(x) \leq \phi(x)$ for all $x \in S$.

Remark 2.4.1 Condition (i) states the convexity of $\phi(x)$, while condition (ii) states that the function $\phi(x)$ underestimates $f(x)$ in the whole domain. Condition (iii) states that there does not exist any convex underestimating function that is tighter than the convex envelope $\phi(x)$.

Theorem 2.4.1 *Let $f(x)$ be a lower semicontinuous function defined on the convex compact set $S \in \Re^n$, and $\phi(x)$ be the convex envelope of $f(x)$ on S. Then,*

(i) $\min_{x \in S} f(x) = \min_{x \in S} \phi(x) = f^*$.

(ii) $\{\boldsymbol{y} \in S : f(\boldsymbol{y}) = f^*\} \subseteq \{\boldsymbol{y} \in S : \phi(\boldsymbol{y}) = f^*\}$

Remark 2.4.2 This theorem states that for each nonconvex optimization problem that has a convex feasible region we can associate a convex optimization problem which has the same optimal solution as the nonconvex optimization problem. Even though this is a powerful theoretical result, note that identifying the convex envelope of arbitrary multivariate functions is a very challenging problem by itself.

2.4.1 Convex Envelope of a Concave Function over a Polytope

The convex envelope of a concave function over a polytope is based on the following theorem :

Theorem 2.4.2 *Let the concave function $f(\boldsymbol{x})$ be defined over a polytope P, and let the vertices of the polytope be $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_m$. Then, the convex envelope $\phi(\boldsymbol{x})$ of the concave function $f(\boldsymbol{x})$ over the polytope P can be given by the following problem :*

$$\begin{aligned} \phi(\boldsymbol{x}) = \min_{\lambda} & \sum_{i=1}^{m} \lambda_i f(\boldsymbol{v}_i) \\ s.t. \quad & \sum_{i=1}^{m} \lambda_i \boldsymbol{v}_i = \boldsymbol{x} \\ & \sum_{i=1}^{m} \lambda_i = 1 \\ \lambda = (\lambda_1, \ldots, \lambda_m) \geq & \mathbf{0} \end{aligned} \tag{2.50}$$

2.4.2 Convex Envelope of a Concave Function over a Simplex

The convex envelope of a concave function over a simplex is an affine function that can be uniquely determined by a system of linear equations as the following theorem states.

Theorem 2.4.3 *Let the concave function $f(\boldsymbol{x})$ be defined over a simplex S, and the vertices of the simplex be $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n$. Then, the convex envelope $\phi(\boldsymbol{x})$ of the concave function $f(\boldsymbol{x})$ over the simplex S is the affine function :*

$$\phi(\boldsymbol{x}) = \boldsymbol{c}^T \boldsymbol{x} + \boldsymbol{b}$$

where the coefficient vectors $\mathbf{c}, \mathbf{b}$ are uniquely determined by the system of linear equations

$$f(\boldsymbol{v}_i) = \boldsymbol{c}^T \boldsymbol{v}_i + \boldsymbol{b} \quad i = 1, \ldots, n.$$

2.4.3 Convex Envelope of a Concave Univariate Function

The convex envelope of a univariate concave function $f(x)$ over $[x^L, x^U]$ is its linearization that goes through the endpoints :

$$f(x^L) + \frac{f(x^U) - f(x^L)}{x^U - x^L}(x - x^L). \tag{2.51}$$

2.4.4 *Convex Envelope of a Bilinear Function*

In the case of a bilinear function xy, McCormick (1976), Al-Khayyal and Falk (1983) showed that the convex envelope over the domain $[x^L, x^U] \times [y^L, y^U]$ is obtained by introducing a new variable w which replaces every occurrence of xy in the problem and satisfies the following relationship:

$$w = \max\{x^L y + y^L x - x^L y^L;\ x^U y + y^U x - x^U y^U\}. \tag{2.52}$$

This lower bound can be relaxed and written as

$$\begin{array}{rcl} w & \geq & x^L y + y^L x - x^L y^L, \\ w & \geq & x^U y + y^U x - x^U y^U. \end{array} \tag{2.53}$$

An upper bound can be imposed on w to construct a better approximation of the original problem (McCormick, 1976). This is achieved through the addition of two linear constraints:

$$\begin{array}{rcl} w & \leq & x^U y + y^L x - x^U y^L, \\ w & \leq & x^L y + y^U x - x^L y^U. \end{array} \tag{2.54}$$

The convex envelope for a bilinear term xy with $x \in [x^L, x^U]$ and $y \in [y^L, y^U]$ is given by Equation (2.52). Note that this is a discontinuous function which is piecewise linear. The following theorem defines the regions of applicability of each linear expression.

Theorem 2.4.4 *The convex envelope of a bilinear term xy with $x \in [x^L, x^U]$ and $y \in [y^L, y^U]$ is*

$$w = \begin{cases} x^L y + y^L x - x^L y^L & \text{if } y \leq -\frac{y^U - y^L}{x^U - x^L}x + \frac{x^U y^U - x^L y^L}{x^U - x^L} \\ x^U y + y^U x - x^U y^U & \text{otherwise} \end{cases}$$

Proof. Using the definition of w given by Equation (2.52), let us determine when $w = x^L y + y^L x - x^L y^L$, that is, when $x^L y + y^L x - x^L y^L \geq x^U y + y^U x - x^U y^U$.

$$\begin{aligned} & x^L y + y^L x - x^L y^L \geq x^U y + y^U x - x^U y^U \\ & \quad \Leftrightarrow \quad -(x^U - x^L)y - (y^U - y^L)x \geq x^L y^L + x^U y^U \\ & \quad \Leftrightarrow \quad y \leq -\frac{y^U - y^L}{x^U - x^L}x + \frac{x^U y^U - x^L y^L}{x^U - x^L} \end{aligned}$$

Geometrical Interpretation. From Theorem 2.4.4, the line separating the two regions of applicability of the linear underestimators is given by

$$y = -\frac{y^U - y^L}{x^U - x^L}x + \frac{x^U y^U - x^L y^L}{x^U - x^L}.$$

This can be equivalently expressed as

$$\begin{aligned} y &= -\frac{y^U - y^L}{x^U - x^L}x + \frac{x^U y^U - x^L y^L}{x^U - x^L} + \frac{x^L y^U - x^L y^U}{x^U - x^L} \\ &= -\frac{y^U - y^L}{x^U - x^L}x + \frac{y^U - y^L}{x^U - x^L}x^L + \frac{x^U y^U - x^L y^U}{x^U - x^L} \\ &= \frac{y^U - y^L}{x^L - x^U}(x - x^L) + y^U \end{aligned}$$

This is the equation of a line passing through (x^L, y^U) and (x^U, y^L). As shown in Figure 2.1, it crosses the rectangle $[x^L, x^U] \times [y^L, y^U]$ diagonally, from the top left corner to the bottom right corner. Region (1) corresponds to the domain of applicability of $x^L y + y^L x - x^L y^L$ and region (2) to that of $x^U y + y^U x - x^U y^U$.

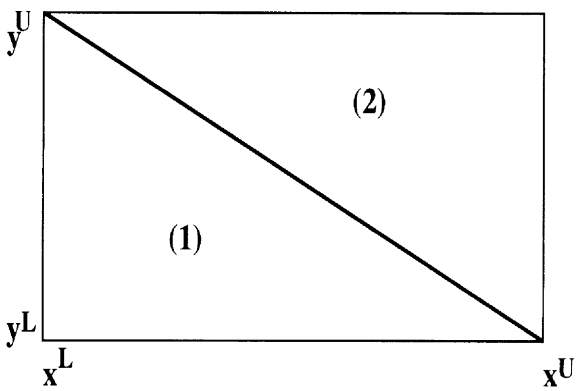


Figure 2.1. Geometrical Interpretation of Theorem 2.4.4.

An important feature of the convex envelope is its maximum separation distance from the actual function that it underestimates. Androulakis *et al.* (1995) determined the maximum separation distance of a bilinear term and its convex envelope. This basic result is represented by the following property :

Property 2.4.1 *The maximum separation distance of the bilinear term xy from its convex envelope,* $\max\left(x^L y + xy^L - x^L y^L,\ x^U y + xy^U - x^U y^U\right)$, *inside the rectangle* $\left[x^L, x^U\right] \times \left[y^L, y^U\right]$ *is equal to one fourth of the area of*

the rectangular domain,

$$\frac{\left(x^U - x^L\right)\left(y^U - y^L\right)}{4}.$$

and occurs at the middle point

$$x^m = \frac{x^L + x^U}{2}, \quad y^m = \frac{y^L + y^U}{2}.$$

Proof. This can be formulated as the following optimization problem.

$$\begin{aligned} \max_{x,y} \quad & xy - \max\left(x^L y + x y^L - x^L y^L,\ x^U y + x y^U - x^U y^U\right) \\ \text{subject to} \quad & x^L \le x \le x^U \\ & y^L \le y \le y^U \end{aligned}$$

By substituting in the objective function $x = x^L$ or $x = x^U$ or $y = y^L$ or $y = y^U$ the maximum separation becomes zero. This implies that for any point in the perimeter of the rectangle $\left[x^L, x^U\right] \times \left[y^L, y^U\right]$ xy matches its convex envelope and thus the point where the maximum separation occurs must be an interior point. After replacing the $\max_{x,y}(\cdot)$ operator with the equivalent $-\min_{x,y} -(\cdot)$ operator and eliminating the max over the two linear cuts at the expense of two new inequality constraints we have:

$$\begin{aligned} -\min_{x,y} \quad & -xy + z \\ \text{subject to} \quad z &\ge x^L y + x y^L - x^L y^L \\ z &\ge x^U y + x y^U - x^U y^U \\ & x^L \le x \le x^U \\ & y^L \le y \le y^U \end{aligned}$$

Let $\mu_1, \mu_2 \ge 0$ be the multipliers associated with the two new inequality constraints. Clearly, the multipliers associated with the variable bound constraints are zero since the solution will be an interior point. The KKT conditions yield the following stationarity conditions:

$$\begin{aligned} \mu_1 + \mu_2 - 1 &= 0 \\ \mu_1 x^L + \mu_2 x^U - x &= 0 \\ \mu_1 y^L + \mu_2 y^U - y &= 0 \\ \left(-z + x^L y + x y^L - x^L y^L\right)\mu_1 &= 0 \\ \left(-z + x^U y + x y^U - x^U y^U\right)\mu_2 &= 0 \\ \mu_1, \mu_2 &\ge 0 \end{aligned}$$

Clearly, at least one of μ_1, μ_2 must be nonzero, leading to the following three cases:

(i) $\mu_1 = 0,\ \mu_2 = 1$

(ii) $\mu_1 = 1,\ \mu_2 = 0$

(iii) $\mu_1 > 0,\ \mu_2 > 0$

If $\mu_1 = 1$ or $\mu_2 = 1$ then we have $-z + x^L y + xy^L - x^L y^L = 0$ or $-z + x^U y + xy^U - x^U y^U = 0$ respectively. Both cases (i) and (ii) lead to a zero maximum separation implying that they correspond to local minima. The single remaining case (iii) yields the following linear system of equations in μ_1, μ_2, x, y, z

$$\begin{aligned}
\mu_1 + \mu_2 - 1 &= 0 \\
\mu_1 x^L + \mu_2 x^U - x &= 0 \\
\mu_1 y^L + \mu_2 y^U - y &= 0 \\
-z + x^L y + xy^L - x^L y^L &= 0 \\
-z + x^U y + xy^U - x^U y^U &= 0
\end{aligned}$$

Solution of this system gives:

$$x = \frac{x^L + x^U}{2},\quad y = \frac{y^L + y^U}{2},\quad z = \frac{x^U y^L + x^L y^U}{2},\quad \mu_1 = \mu_2 = \frac{1}{2}$$

The maximum separation therefore is

$$xy - z = \frac{\left(x^U - x^L\right)\left(y^U - y^L\right)}{4}.$$

2.4.5 *Convex Envelope of Products of Univariate Functions*

For products of univariate functions that are twice continuously differentiable, the following two theorems (Maranas and Floudas (1995)) provide the basis for the underestimation and establish the convex envelope.

Theorem 2.4.5 *Let f, g be twice continuously differentiable univariate functions $f(x), g(y) \in C^2$ defined inside a rectangle $\left[\left(x^L, x^U\right), \left(y^L, y^U\right)\right]$ and*

$$l(x,y) = \max\left\{\ \phi\left(f^L g(y)\right) + \phi\left(g^L f(x)\right) - f^L g^L,\ \phi\left(f^U g(y)\right) + \phi\left(g^U f(x)\right) - f^U g^U\right\}$$

$$\begin{aligned}
\textit{where}\ f^L &= \inf_{x^L \le x \le x^U} f(x), \\
f^U &= \sup_{x^L \le x \le x^U} f(x), \\
g^L &= \inf_{x^L \le x \le x^U} g(x), \\
g^U &= \sup_{x^L \le x \le x^U} g(x)
\end{aligned}$$

and $\phi\left(f^L g(y)\right), \phi\left(g^L f(x)\right), \phi\left(f^U g(y)\right)$, *and* $\phi\left(g^U f(x)\right)$ *are the convex envelopes of the univariate functions* $f^L g(y), g^L f(x), f^U g(y)$, *and* $g^U f(x)$, *respectively. Then we have that:*

(i) $l(x,y)$ *is convex,* $\forall\,(x,y) \in \left[x^L, x^U\right] \times \left[y^L, y^U\right]$.

(ii) $f(x)g(y) \geq l(x,y)$, $\forall\,(x,y) \in \left[x^L, x^U\right] \times \left[y^L, y^U\right]$.

Proof. Both functions $\phi\left(f^L g(y)\right) + \phi\left(g^L f(x)\right) - f^L g^L$ and $\phi\left(f^U g(y)\right) + \phi\left(g^U f(x)\right) - f^U g^U$ are convex as the sum of the convex envelopes of univariate functions. Since the maximum of two convex functions is a convex function as well, statement (i) is true and $l(x,y)$ is convex for all (x,y) in $\left[x^L, x^U\right] \times \left[y^L, y^U\right]$.

Because,

$$f(x) - f^L \geq 0,\ \forall\, x \in \left[x^L, x^U\right] \text{ and } g(y) - g^L \geq 0,\ \forall\, y \in \left[y^L, y^U\right]$$

we have

$$\left(f(x) - f^L\right)\left(g(y) - g^L\right) \geq 0,\ \forall\,(x,y) \in \left[x^L, x^U\right] \times \left[y^L, y^U\right].$$

After rearranging terms we obtain,

$$f(x)g(y) \geq f^L g(y) + f(x)g^L - f^L g^L,\ \ \forall\,(x,y) \in \left[x^L, x^U\right] \times \left[y^L, y^U\right].$$

By the definition of the convex envelope we know that,

$$\begin{aligned} f^L g(y) &\geq \phi(f^L g(y)),\ \forall\, y \in \left[y^L, y^U\right] \text{ and} \\ g^L f(x) &\geq \phi(g^L f(x)),\ \forall\, x \in \left[x^L, x^U\right]. \end{aligned}$$

Therefore,

$$f(x)g(y) \geq \phi(f^L g(y)) + \phi(f(x)g^L) - f^L g^L,\ \forall\,(x,y) \in \left[x^L, x^U\right] \times \left[y^L, y^U\right]. \tag{2.55}$$

Furthermore, by following the same line of reasoning on the relation,

$$\left(f(x) - f^U\right)\left(g(y) - g^U\right) \geq 0,\ \forall\,(x,y) \in \left[x^L, x^U\right] \times \left[y^L, y^U\right],$$

we obtain:

$$f(x)g(y) \geq \phi(f^U g(y)) + \phi(g^U f(x)) - f^U g^U,\ \forall\,(x,y) \in \left[x^L, x^U\right] \times \left[y^L, y^U\right]. \tag{2.56}$$

Relations (2.55) and (2.56) imply that statement (ii) is true.

Based on Theorem 2.4.5 the function $l(x,y)$ can be utilized as a tight convex lower bounding function of the product of two continuous and twice differentiable functions. It can also be shown that under certain conditions $l(x,y)$

corresponds to the actual convex envelope of the product $f(x)g(y)$. These conditions are stated in the following theorem:

Theorem 2.4.6 *Let (i) the univariate functions,*

$$g^L f(x),\ g^U f(x) \text{ and } f^L g(y),\ f^U g(y)$$

be concave in $[x^L, x^U]$ *and* $[y^L, y^U]$*, respectively; and (ii) the functions* $f(x)$*,* $g(y)$ *be monotonic. Then,* $l(x,y)$ *is the convex envelope of* $f(x)g(y)$*, that is:*

$$\phi(f(x)g(y)) = l(x,y),\ \forall\,(x,y) \in [x^L, x^U] \times [y^L, y^U].$$

Proof. Theorem 2.4.5 proves that function $l(x,y)$ conforms with Properties (i) and (ii) of the definition of the convex envelope. Therefore, it remains to show that it satisfies Property (iii) of the convex envelope definition. Because the convex envelope of a univariate concave function defined in an interval is the line segment connecting the two end points we have:

$$\begin{aligned}
\phi(g^L f(x)) &= g^L\left[\frac{f(x^U)-f(x^L)}{x^U-x^L}x + \frac{x^U f(x^L) - x^L f(x^U)}{x^U - x^L}\right]\\
\phi(g^U f(x)) &= g^U\left[\frac{f(x^U)-f(x^L)}{x^U-x^L}x + \frac{x^U f(x^L) - x^L f(x^U)}{x^U - x^L}\right]\\
\phi(f^L g(y)) &= f^L\left[\frac{g(y^U)-g(y^L)}{y^U-y^L}x + \frac{y^U g(y^L) - y^L g(y^U)}{y^U - y^L}\right]\\
\phi(f^U g(y)) &= f^U\left[\frac{g(y^U)-g(y^L)}{y^U-y^L}x + \frac{y^U g(y^L) - y^L g(y^U)}{y^U - y^L}\right]
\end{aligned}$$

After substituting these expressions into the relation for $l(x,y)$ we obtain:

$$l(x,y) = \max\left(l_1(x,y), l_2(x,y)\right)$$

$$\begin{aligned}
l_1(x,y) &= \left[g^L\frac{f(x^U)-f(x^L)}{x^U-x^L}\right]x + \left[f^L\frac{g(y^U)-g(y^L)}{y^U-y^L}\right]y\\
&+ \left[g^L\frac{x^U f(x^L) - x^L f(x^U)}{x^U-x^L} + f^L\frac{y^U g(y^L) - y^L g(y^U)}{y^U-y^L} - f^L g^L\right]
\end{aligned}$$

$$\begin{aligned}
l_2(x,y) &= \left[g^U\frac{f(x^U)-f(x^L)}{x^U-x^L}\right]x + \left[f^U\frac{g(y^U)-g(y^L)}{y^U-y^L}\right]y\\
&+ \left[g^U\frac{x^U f(x^L) - x^L f(x^U)}{x^U-x^L} + f^U\frac{y^U g(y^L) - y^L g(y^U)}{y^U-y^L} - f^U g^U\right]
\end{aligned}$$

Because $f(x), g(y)$ are monotonic one of the following alternatives is true:

$$\begin{array}{llllllllll}
(a) & f(x^L) = f^L, & f(x^U) = f^U, & g(x^L) = g^L, & g(x^U) = g^U\\
(b) & f(x^L) = f^L, & f(x^U) = f^U, & g(x^L) = g^U, & g(x^U) = g^L\\
(c) & f(x^L) = f^U, & f(x^U) = f^L, & g(x^L) = g^L, & g(x^U) = g^U\\
(d) & f(x^L) = f^U, & f(x^U) = f^L, & g(x^L) = g^U, & g(x^U) = g^L
\end{array}$$

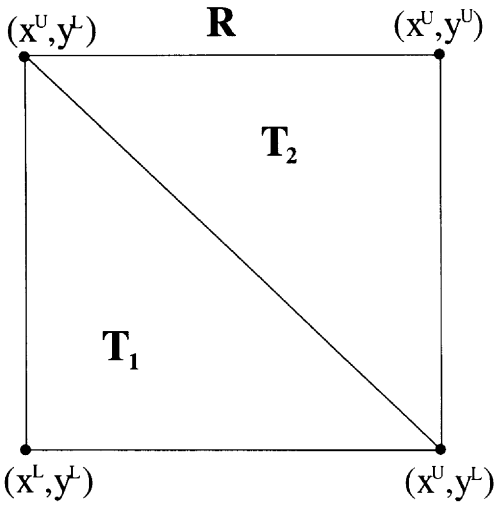


Figure 2.2. Decomposition of rectangle R into two triangles T_1, T_2.

Assuming that (a) is true we have:

$$\begin{array}{rcl} l_1(x^L,y^L) & = & f(x^L)g(y^L) \\ l_1(x^L,y^U) & = & f(x^L)g(y^U) \\ l_1(x^U,y^L) & = & f(x^U)g(y^L) \end{array} \quad \text{and} \quad \begin{array}{rcl} l_2(x^U,y^U) & = & f(x^U)g(y^U) \\ l_2(x^L,y^U) & = & f(x^L)g(y^U) \\ l_2(x^U,y^L) & = & f(x^U)g(y^L) \end{array}$$

This implies that the original rectangle

$$\mathcal{R} = \left[(x^L,y^L),(x^L,y^U),(x^U,y^L),(x^U,y^U)\right]$$

naturally decomposes into the following two disjoint triangles,

$$\mathcal{T}_1 = \left[(x^L,y^L),(x^L,y^U),(x^U,y^L)\right], \text{ and } \mathcal{T}_2 = \left[(x^U,y^U),(x^L,y^U),(x^U,y^L)\right]$$

at whose vertices the linear functions $l_1(x,y), l_2(x,y)$ match the original function $f(x)g(y)$ respectively (See Figure 2.2). If $l(x,y)$ were not the convex envelope of $f(x)g(y)$ over the rectangular domain $\mathcal{R}$ then, there would be a third convex function $l_3(x,y)$ underestimating $f(x)g(y)$ over $\mathcal{R}$ and a point $(\bar{x},\bar{y}) \in \mathcal{R}$ such that:

$$l(\bar{x},\bar{y}) < l_3(\bar{x},\bar{y})$$

Suppose that $(\bar{x},\bar{y}) \in \mathcal{T}_1$. Then $(\bar{x},\bar{y})$ is a unique convex combination of the three extreme points v^1, v^2, v^3 of $\mathcal{T}_1$. Hence, for the affine function l, there

exists unique positive $\lambda_1, \lambda_2, \lambda_3$ satisfying $\sum_{i=1}^{3} \lambda_i = 1$ such that

$$l(\bar{x}, \bar{y}) \;=\; l\left(\sum_{i=1}^{3} \lambda_i v^i\right) \;=\; \sum_{i=1}^{3} \lambda_i l(v^i) \;=\; l(\bar{x}, \bar{y}).$$

Because l_3 is the convex envelope of $f(x)g(y)$ inside $\mathcal{R}$, (i) l_3 is convex and (ii) it matches $f(x)g(y)$ at all vertex points like $l(x, y)$ does which implies:

$$l_3(\bar{x}, \bar{y}) \;=\; l_3\left(\sum_{i=1}^{3} \lambda_i v^i\right) \;\leq\; \sum_{i=1}^{3} \lambda_i l_3(v^i) \;=\; \sum_{i=1}^{3} \lambda_i l(v^i) \;=\; l(\bar{x}, \bar{y}).$$

This contradicts the initial hypothesis $l(\bar{x}, \bar{y}) < l_3(\bar{x}, \bar{y})$ and therefore, $l(x, y)$ is indeed the convex envelope of $f(x)g(y)$ in $\mathcal{R}$. Note that a similar argument holds if $(\bar{x}, \bar{y}) \in \mathcal{T}_2$. Moreover, depending on which monotonicity combination (a), (b) ,(c) or (d) is true it is always possible to partition $\mathcal{R}$ into two triangles $\mathcal{T}_1, \mathcal{T}_2$ by halving along one of the diagonals. Therefore, by following the same line of thought for combinations (b), (c) and (d) it is straightforward to extend this proof for all monotonicity combinations.

The analysis for the convex lower bounding of products of two univariate functions can be extended to accommodate the product of N univariate functions. This is accomplished by performing convex lower bounding for pairs of univariate functions in a recursive manner until no pairs are left. One of the possible alternatives of combining pairs is to start with convex lower bounding the last two functions of the product and next recursively convex lower bound the resulting function with the currently last term in the product. This analysis is presented in Chapter 23 (see Theorem 23.3). Based on this analysis valid underestimators can be obtained for trilinear, fractional, and fractional trilinear terms. These underestimators are presented in Chapter 10 along with theoretical results on the maximum separation distance.

2.4.6 *Convex Envelope of a General Univariate Function*

In this section, we discuss the procedure suggested by Maranas and Floudas (1995) for determining the convex envelope of a general twice differentiable univariate function.

Computing the convex envelopes of arbitrary twice differentiable functions in a single variable appears frequently as a task in many complex convex lower bounding situations. In some cases, this is straightforward, for example if $f \in C^2 : [a, b] \to \mathcal{R}$ is convex then its convex envelope coincides with the original function:

$$\phi(f(x)) \;=\; f(x), \; \forall\, x \;\in\; [a, b] \;\text{ if and only if }\; f(x) \text{ is convex in } [a, b]\,.$$

If $f(x)$ is concave, then its convex envelope is a line segment connecting the end points of the graph of the function:

$$\phi(f(x)) \;=\; \frac{f(b) - f(a)}{b - a} x + \frac{b f(a) - a f(b)}{b - a}, \; \forall\, x \;\in\; [a, b]\,.$$

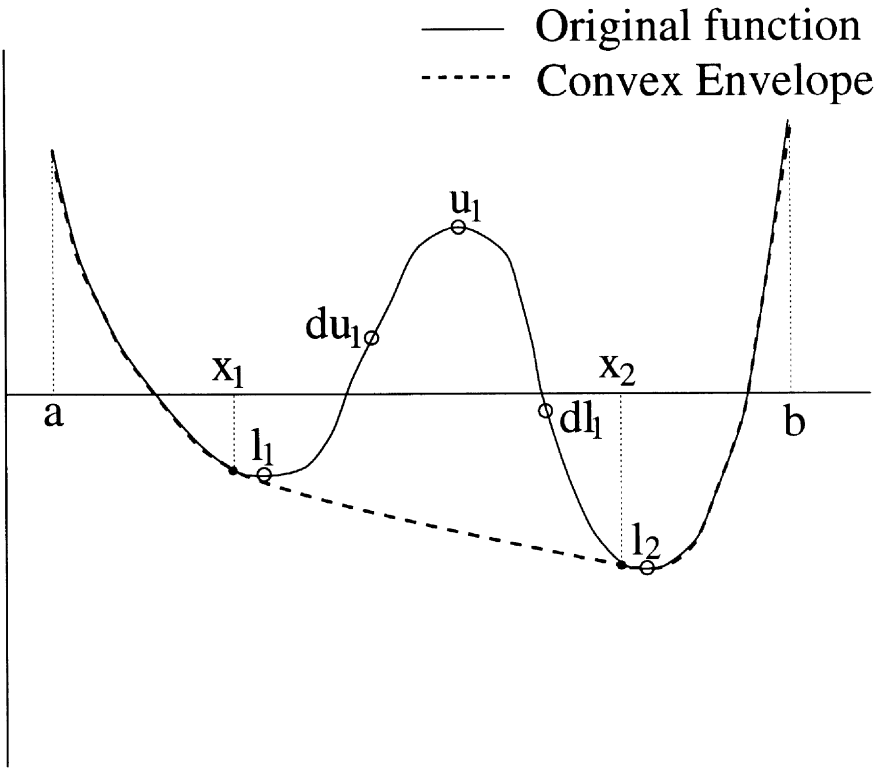


Figure 2.3. Convex envelope of a general univariate function.

Constructing the convex envelope of an arbitrary nonconvex function, however, is a much more demanding task because its graph alternates between convex and concave portions. In general, the convex envelope of nonconvex univariate functions is composed by different representations in different subintervals. More specifically, the convex envelope curve alternates between the original function (convex portions of the curve) and line segments (concave portions) (See Figure 2.3). The challenge here is to locate the exact points where the convex envelope changes representation from a line segment to trace the curve of the original function and vice–versa. The number of these "switch–over" points depends on the frequency with which $f''(x)$ changes sign in the interval

$[a, b]$. The actual locations of these points depend not only on the shape of the function but also on the location of the corner points.

Locating the exact location and number of points requires knowledge of global information about the univariate function $f(x)$ in the interval $[a, b]$. More specifically, the location of all unconstrained local minima l_i, local maxima u_i, and inflection points dl_i, du_i is needed:

$$\begin{array}{rlll} l_i: & f'(l_i) = 0, & f''(l_i) \geq 0 \\ u_i: & f'(u_i) = 0, & f''(u_i) \leq 0 \\ dl_i: & f''(dl_i) = 0, & f'(dl_i) \leq 0 \\ du_i: & f''(du_i) = 0, & f'(du_i) \geq 0 \end{array}$$

These points can be obtained by utilizing a robust solver guaranteed to locate all solutions of univariate functions in a interval (Hansen, 1992). Due to the alternating of convex and concave portions of the nonconvex function f, there is a specific order with which these points appear in the graph of the univariate function $f(x)$ which is:

$$\cdots [dl - l - du - u]_i \cdots$$

This naturally provides a partitioning of the initial interval $[a, b]$ into convex subintervals $[dl_i, du_i]$ and concave ones $[du_i, dl_{i+1}]$.

The procedure for locating the first point where the convex envelope changes representation depends on whether $f(x)$ is convex or concave at $x = a$. If f is concave at $x = a$ then the initial segment of the convex envelope is a line. The next segment of the convex envelope is the function itself starting at the point x where the slope f' of f equals the slope of the line connecting a with x.

$$\frac{f(x) - f(a)}{x - a} = f'(x)$$

Note that x belongs to one of the convex subintervals $[dl_i, du_i]$ since f must be convex at x. This implies that the task of locating x corresponds to drawing one tangent from the fixed point $(a, f(a))$ to each one of the convex function representations defined in the subintervals $[dl_i, du_i]$. Because there exists a single tangent to a convex function drawn from a point outside a convex function (Luenberger, 1984) any standard bisection algorithm can be utilized to locate x. The correct subinterval $[dl_i, du_i]$ is then the one which provides a line that does not cut-off any portion of the curve $f(x)$.

If now, f is convex at $x = a$ then the initial segment of the convex envelope can be either a line or the function itself. If there exists a convex subinterval $[dl_i, du_i]$, $i = 2, \ldots$ where the equation $f(x) - f(a) = f'(x)(x - a)$ has a solution x which defines a line that does not cut-off any portion of the curve $f(x)$, then the initial segment of the convex envelope is a line connecting the points $(a, f(a))$ and $(x, f(x))$. Otherwise, the initial segment of the convex envelope is the function f itself. The last point of this segment x_1 is found by locating the end points x_1, x_2 of the next subinterval where the convex envelope

becomes a line segment. This corresponds to drawing a common tangent to f inside the intervals $[dl_1, du_1]$ and $[dl_i, du_i]$, $i = 2, \ldots$ and is the solution of the following system of two equations:

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

where $x_1 \in [dl_1, du_1]$ and $x_2 \in [dl_i, du_i]$, $i = 2, \ldots$. Again, the correct subinterval $[dl_i, du_i]$, $i = 2, \ldots$ is then the one for which the line connecting the points $(x_1, f(x_1))$ and $(x_2, f(x_2))$ does not cut-off any portion of the curve $f(x)$. The next line segment is then found by iteratively solving the system of two equations for locating the new points x_1, x_2. This time however, $x_1 \in [dl_i, du_i]$ and $x_2 \in [dl_j, du_j]$, $j = i + 1, \ldots$. This is continued until the end point $x = b$ is met. Based on this analysis an iterative systematic procedure is defined for constructing the convex envelope of arbitrary univariate nonconvex functions.

2.5 LOCAL, GLOBAL, ϵ-GLOBAL MINIMA

This section discusses the definitions of the local, global and ϵ-global mimima.

Consider the problem of minimizing $f(\boldsymbol{x})$ subject to $\boldsymbol{x} \in S$, where the set S is defined as :

$$S = \{\boldsymbol{x} \in X \mid \boldsymbol{h}(\boldsymbol{x}) = \boldsymbol{0},\ \boldsymbol{g}(\boldsymbol{x}) \leq \boldsymbol{0}\},$$

where X is a closed convex set.

Definition 2.5.1 (Feasible Solution) *A point $\boldsymbol{x} \in S$ is a feasible solution to this problem.*

Definition 2.5.2 (Local Minimum) *Suppose that $\boldsymbol{x}^\star \in S$ and that there exists an $\epsilon > 0$ such that*

$$f(\boldsymbol{x}) \geq f(\boldsymbol{x}^\star) \qquad \forall\, \boldsymbol{x} \in S \ : \ \| \boldsymbol{x} - \boldsymbol{x}^\star \| < \epsilon \ ,$$

then, $\boldsymbol{x}^\star$ is a local minimum.

Definition 2.5.3 (Global Minimum) *Suppose that $\boldsymbol{x}^\star \in S$ and*

$$f(\boldsymbol{x}) \geq f(\boldsymbol{x}^\star) \qquad \forall\, \boldsymbol{x} \in S \ ,$$

then, $\boldsymbol{x}^\star$ is a global minimum.

Definition 2.5.4 (Unique Global Minimum) *Suppose that $\boldsymbol{x}^\star \in S$ and*

$$f(\boldsymbol{x}) > f(\boldsymbol{x}^\star) \qquad \forall\, \boldsymbol{x} \in S \ ,$$

then, $\boldsymbol{x}^\star$ is the unique global minimum.

Theorem 2.5.1 *Let S be a non-empty convex set in $\Re^n$ and $\boldsymbol{x}^\star \in S$ be a local minimum.*

(i) *If $f(\boldsymbol{x})$ is convex, then $\boldsymbol{x}^\star$ is a global minimum.*

(ii) *If $f(\boldsymbol{x})$ is strictly convex, then $\boldsymbol{x}^\star$ is the unique global minimum.*

In many practical applications the determination of the exact global minimum may be either very expensive computationally or unnecessary. Most often we are interested in identifying the global minimum within some predescribed tolerance $\epsilon \geq 0$. The following definition of an ϵ-global minimum is therefore introduced.

Definition 2.5.5 (ϵ-Global Minimum) *Suppose that $\boldsymbol{x}^\star \in S$ is a feasible solution, $\epsilon \geq 0$ is a small predescribed tolerance, and*

$$f(\boldsymbol{x}) \geq f(\boldsymbol{x}^\star) - \epsilon \qquad \forall\, \boldsymbol{x} \in S\ ,$$

then, $\boldsymbol{x}^\star$ is an ϵ-global minimum.

2.6 DIFFERENCE OF CONVEX (D.C.) FUNCTIONS

The d.c. functions play an important role in global optimization and they have been studied extensively especially through the pioneering work of Tuy (Tuy (1998), Horst and Tuy (1990), Horst *et al.* (1995), Konno *et al.* (1997)). This section (i) introduces the definitions of the d.c. functions and locally d.c. functions, (ii) presents the key properties of the d.c. functions, and (iii) discusses the representation of several classes of functions into the d.c. form.

2.6.1 Basic Definitions

Definition 2.6.1 (D.C. Function) *Let the real valued functions $f(\boldsymbol{x})$, $p(\boldsymbol{x})$, $q(\boldsymbol{x})$ defined on a convex set $S \subset \Re^n$, and let $p(\boldsymbol{x})$ and $q(\boldsymbol{x})$ be convex. If, for all $\boldsymbol{x} \in S$, the function $f(\boldsymbol{x})$ can be expressed as :*

$$f(\boldsymbol{x}) \;=\; p(\boldsymbol{x}) - q(\boldsymbol{x})$$

then $f(\boldsymbol{x})$ is called d.c. on S.

Definition 2.6.2 (Locally D.C. Function) *Let the real valued function $f(\boldsymbol{x})$ be defined in $\Re^n$, and let $B(\boldsymbol{x}_0, \epsilon)$ be a ball defined as*

$$B(\boldsymbol{x}_0, \epsilon) \;=\; \{\boldsymbol{x} \in \Re^n \;:\; \|\boldsymbol{x} - \boldsymbol{x}_0\| \;\leq\; \epsilon\},\ \epsilon \;\geq\; 0.$$

If, for all $\boldsymbol{x}_0 \in \Re^n$, there exists a ball $B(\boldsymbol{x}_0, \epsilon)$ such that the function $f(\boldsymbol{x})$ is d.c. on $B(\boldsymbol{x}_0, \epsilon)$, then the function $f(\boldsymbol{x})$ is called locally d.c..

2.6.2 Properties of D.C. Functions

D.C. functions exhibit an important set of properties which are outlined in the following :

(i) Let $f_1(\boldsymbol{x}), \ldots, f_n(\boldsymbol{x})$ be d.c. functions on a convex subset S of $\Re^n$, and $\lambda_1, \ldots, \lambda_n$, be any real numbers. Then, the following functions are d.c. :

$$\lambda_1 f_1(\boldsymbol{x}) + \ldots + \lambda_n f_n(\boldsymbol{x})$$

$$\max\{f_1(\boldsymbol{x}), \ldots, f_n(\boldsymbol{x})\}$$

$$\min\{f_1(\boldsymbol{x}), \ldots, f_n(\boldsymbol{x})\}$$

$$\prod_i f_i(\boldsymbol{x}).$$

(ii) Let $f(\boldsymbol{x})$ be a d.c. function. Then, the following functions are d.c. :

$$\max\{0, f(\boldsymbol{x})\}$$

$$\min\{0, f(\boldsymbol{x})\}$$

$$|f(\boldsymbol{x})|.$$

(iii) Let S be a convex, open or closed set. Then, any locally d.c. function on S is also a d.c. function on S.

(iv) A C^2 (twice continuously differentiable) function is a d.c. function on any compact convex set $S \in \Re^n$.

(v) Let $f(\boldsymbol{x}) : \Re^n \to \Re$ be a d.c. function and $g(y)$ be a convex function $\Re \to \Re$. Then, the composite function $g(f(\boldsymbol{x}))$ is a d.c. function.

(vi) Let $S \in \Re^n$ be a nonempty closed set and the distance function $d_S(\boldsymbol{x})$ be defined by :

$$d_S(\boldsymbol{x}) = \inf_{\boldsymbol{y} \in S}\{\|\boldsymbol{y} - \boldsymbol{x}\|\}.$$

Then, the square of the distance function, $d_S^2(\boldsymbol{x})$ is a d.c. function.

(vii) A continuous piece-wise linear function is a d.c. function.

(viii) Any polynomial function in $\Re^n$ is a d.c. function in $\Re^n$.

(viiii) Any signomial function (i.e., difference of two posynomials) in $\Re^n$ is a d.c. function in $\Re^n$.

(x) Any factorable function $f(\boldsymbol{x})$ (McCormick (1972),(1983)), that is a function that can be constructed after the following sequence of operations :

$$f_i(\boldsymbol{x}) \quad = \quad x_i \;\; i = 1, 2, \ldots, n$$

and for $k > n$, the functions f_k become :

$$\begin{aligned} f_k(\boldsymbol{x}) &= f_l(\boldsymbol{x}) + f_m(\boldsymbol{x}) \quad \text{for some } l, m < k \\ f_k(\boldsymbol{x}) &= f_l(\boldsymbol{x}) \times f_m(\boldsymbol{x}) \quad \text{for some } l, m < k \\ f_k(\boldsymbol{x}) &= F(f_m(\boldsymbol{x})) \quad \text{for some } m < k \end{aligned}$$

where the function F is a simple d.c. function of one variable (e.g., $F(y) = y^a$; $F(y) = e^y$, $F(y) = cos(y)$), is a d.c. function.

(xi) The directional derivative $f'(\boldsymbol{x}; \mathbf{d})$ of a d.c. function $f(\boldsymbol{x}) = p(\boldsymbol{x}) - q(\boldsymbol{x})$ exists at each $\boldsymbol{x} \in \Re^n$ and in each direction $\mathbf{d} \in \Re^{\mathbf{n}}$, and it is a d.c. function.

2.6.3 Representation of D.C. Functions

A d.c. function can be written as a difference of two convex functions in several ways, and hence it is of primary importance to identify its effective d.c. representation. In this section, we discuss a number of theoretical properties that lead into specific types of d.c. representations. In particular, we focus on (i) separable functions, (ii) products of convex functions, (iii) indefinite quadratic functions, (iv) general twice continuously differentiable (C^2) functions, and (v) composite functions.

2.6.3.1 Separable Functions. A separable function $f(\boldsymbol{x})$ can be written as:

$$f(\boldsymbol{x}) = \sum_{i=1}^{n} f_i(x_i)$$

where the functions $f_i(x_i)$ are univariate functions in x_i. It is apparent that a separable function $f(\boldsymbol{x})$ is d.c. if the univariate functions $f_i(x_i)$ are d.c. functions. As a result, the conditions for a d.c. representation of a separable function rely upon the conditions for a d.c. representation of univariate functions. The following property represents the fundamental result for the d.c. representation of a univariate function (Roberts and Varberg (1973)).

Property 2.6.1 *A univariate function $f(x) : [0, a] \to \Re$ can be represented in a d.c. form, that is $f(x) = p(x) - q(x)$, where $p(x)$ and $q(x)$ are convex and $p'_+(0), p'_-(a), q'_+(0), q'_-(a)$ are all finite, if and only if*

$$f(x) = f(0) + \int_0^x r(t)dt$$

for some function of bounded variation $r(t) : [0, a] \to \Re$.

Remark 2.6.1 Since the function $r(t)$ is of bounded variation, then it can be decomposed into a difference of two monotone nondecreasing functions, that is

$r(t) = r_1(t) - r_2(t)$ where $r_1(t)$ is the total variation of $r(t)$ in the interval $[0,x]$. Therefore, the d.c. representation of the function $f(x)$ takes the form :

$$f(x) = \{f(0) + \int_0^x r_1(t)dt\} - \int_0^x r_2(t)dt$$

Several application areas have models that are continuous piecewise linear and hence the d.c. representation of such functions is important. The following property presents the basis for a d.c. representation of continuous piecewise linear functions.

Property 2.6.2 *Let the continuous piecewise linear function $f(x) : [0,a] \to \Re$, and the sequence of breakpoints $[0, c_1, c_2, \ldots, c_k, a]$. Let the intervals $U_i = [c_i, c_{i+1}]$ with $c_0 = 0$ and $c_{k+1} = a$. Consider the convex functions $w_i(x) : [0,a] \to \Re$, where $i \in I = \{i | f(x)$ is concave at breakpoint $c_i\}$, that agree with $-f(x)$ at every $x \in U_i$, and they are linear on each interval $[0, c_i]$ and $[c_i, a]$. Then, a d.c. representation of $f(x)$ is $f(x) = p(x) - q(x)$, where $p(x)$ is convex and $q(x)$ takes the form :*

$$q(x) = \sum_{i \in I} w_i(x).$$

Remark 2.6.2 This property can be extended to the case of piecewise convex-concave functions such as S-shaped functions (Tuy, 1998).

2.6.3.2 Products of Convex Functions. Products of convex functions appear frequently in applications. The following property presents one way of writing the product of two convex functions in a d.c. representation.

Property 2.6.3 *Let the convex functions $f_1(\boldsymbol{x})$, $f_2(\boldsymbol{x})$ be defined on $\Re^n_+$. A d.c. representation of their product is :*

$$f_1(\boldsymbol{x})f_2(\boldsymbol{x}) = 0.5[f_1(\boldsymbol{x}) + f_2(\boldsymbol{x})]^2 - 0.5[f_1^2(\boldsymbol{x}) + f_2^2(\boldsymbol{x})]$$

Remark 2.6.3 Application of the above property in a recursive way can result in d.c. representations of products of several functions, as well as polynomials (Konno *et al.* (1997)).

2.6.3.3 Indefinite Quadratic Functions. The most challenging classes of quadratic programming problems are those that exhibit an indefinite quadratic function. The following property provides the foundation for a d.c. representation of an indefinite quadratic function.

Property 2.6.4 *Let the indefinite quadratic function $f(\boldsymbol{x})$ be of the form $f(\boldsymbol{x}) = \boldsymbol{x}^T Q \boldsymbol{x}$, where Q is a symmetric indefinite matrix with spectral radius $\rho(Q)$. Then, the function $g(\boldsymbol{x})$ of the form :*

$$g(\boldsymbol{x}) = f(\boldsymbol{x}) + \lambda \|\boldsymbol{x}\|^2$$

is convex for any $\lambda \geq \rho(Q)$.

Remark 2.6.4 Since $g(\boldsymbol{x})$ is convex based on the above property, and since $\lambda||\boldsymbol{x}||^2$ is convex for any $\lambda \geq \rho(Q)$, then the indefinite quadratic function $f(\boldsymbol{x})$ can be written in a d.c. form.

2.6.3.4 General C^2 Functions. General twice continuously differentiable C^2 functions are the dominant type in all applications areas in science, applied science, and engineering. Therefore, the derivation of effective d.c. representations of general C^2 functions are of utmost importance. A very effective d.c. representation of C^2 functions that appear in unconstrained and constrained nonlinear optimization problems is based on the αBB type of underestimators that are described in detail in Chapters 10, 11 and 12 (see also the publications Maranas and Floudas (1994a), Maranas and Floudas (1994b), Androulakis *et al.* (1995), Adjiman and Floudas (1996), Adjiman *et al.* (1998a), Adjiman *et al.* (1998b)). The following key property presents the d.c. representation of general C^2 functions.

Property 2.6.5 *Let the function* $f(\boldsymbol{x})$ *be in* C^2, *and the function* $\mathcal{L}(\boldsymbol{x})$ *defined as :*

$$\begin{aligned} \mathcal{L}(\boldsymbol{x}) &= f(\boldsymbol{x}) + \alpha \sum_{i=1}^{n} (x_i^L - x_i)(x_i^U - x_i) \\ \text{where} \qquad \alpha &\geq \max \left\{ \max_{\substack{i \\ x_i^L \leq x_i \leq y_i^U}} 0, \left(-\frac{1}{2}\lambda_i^f \right) \right\} \end{aligned}$$

where λ_i^f *are the eigenvalues of the hessian matrix of the function* f. *Then, the function* $\mathcal{L}(\boldsymbol{x})$ *is convex.*

Remark 2.6.5 Based on the above property, the d.c. representation of the function $f(\boldsymbol{x}) \in C^2$ takes the form :

$$\begin{aligned} f(\boldsymbol{x}) &= \mathcal{L}(\boldsymbol{x}) - \alpha \sum_{i=1}^{n} (x_i^L - x_i)(x_i^U - x_i) \\ \text{where} \qquad \alpha &\geq \max \left\{ \max_{\substack{i \\ x_i^L \leq x_i \leq y_i^U}} 0, \left(-\frac{1}{2}\lambda_i^f \right) \right\} \end{aligned}$$

Note that the function that is subtracted from $\mathcal{L}(\boldsymbol{x})$ is a convex quadratic function that is separable in each variable x_i. Note also that this type of d.c. representation corresponds to introducing a single α for all the variables which corresponds to the uniform diagonal shift approaches presented in Chapters 10, 11 and 12. It is also possible to develop a d.c. representation with different α's, that is we introduce an α_i for each variable x_i, and this corresponds to the case of non-uniform diagonal shift approaches discussed in Chapter 12. Chapters 10,

11 and 12 present a variety of methods for the calculation of rigorous bounds on the values of the α's, while Chapter 13 presents extensive computational results on several applications.

2.6.3.5 Composite Functions. Composite convex or concave functions with other convex or concave functions appear in a variety of applications that range from location theory to bilevel programming to economics to quadratic assignment to structure prediction of molecules and peptides. Based on the Property (v) of d.c. functions described in section 2.6.2, these composite functions are d.c. functions, and hence their d.c. representation is very useful. The following properties provide the d.c. representation of a convex monotone function of a d.c. function on a compact convex set.

Property 2.6.6 *Let S be a compact convex subset of $\Re^n$, $f(\boldsymbol{x}) : S \to \Re_+$ be a convex function, $g(\boldsymbol{x})$ be a convex function, $q(y) : \Re_+ \to \Re$ be a convex nonincreasing function such that $q'_+(0) > -\infty$, and K is a positive constant such that $K \geq |q'_+(0)|$, where $q'_+(y)$ is the right derivative of $q(y)$ at point y. Then, the function $q(f(\boldsymbol{x}))$ defined as :*

$$q(f(\boldsymbol{x})) = g(\boldsymbol{x}) - Kf(\boldsymbol{x})$$

is a d.c. function on S.

Property 2.6.7 *Let S be a compact convex subset of $\Re^n$, $f(\boldsymbol{x}) = f_1(\boldsymbol{x}) - f_2(\boldsymbol{x})$ where $f_1(\boldsymbol{x}), f_2(\boldsymbol{x}) : S \to \Re_+$ are convex functions such that $f(\boldsymbol{x}) \geq 0, \forall \boldsymbol{x} \in S$, $g(\boldsymbol{x})$ be a convex function, $q(y) : \Re_+ \to \Re$ be a convex nonincreasing function such that $q'_+(0) > -\infty$, and K is a positive constant such that $K \geq |q'_+(0)|$, where $q'_+(y)$ is the right derivative of $q(y)$ at point y. Then, the function $q(f(\boldsymbol{x}))$ defined as :*

$$q(f(\boldsymbol{x})) = g(\boldsymbol{x}) - K[f_1(\boldsymbol{x}) + f_2(\boldsymbol{x})]$$

is a d.c. function on S.

Property 2.6.8 *Let S be a compact convex subset of $\Re^n$, $f(\boldsymbol{x}) = f_1(\boldsymbol{x}) - f_2(\boldsymbol{x})$ where $f_1(\boldsymbol{x}), f_2(\boldsymbol{x}) : S \to \Re_+$ are convex functions such that $0 \leq f(\boldsymbol{x}) \leq a, \forall \boldsymbol{x} \in S$, $g(\boldsymbol{x})$ be a convex function, $q(y) : [0, a] \to \Re$ be a convex nondecreasing function such that $q'_-(a) < +\infty$, and K is a constant such that $K \geq |q'_-(a)|$, where $q'_-(y)$ is the left derivative of $q(y)$ at point y. Then, the function $q(f(\boldsymbol{x}))$ defined as :*

$$q(f(\boldsymbol{x})) = g(\boldsymbol{x}) - K[a + f_2(\boldsymbol{x}) - f_1(\boldsymbol{x})]$$

is a d.c. function on S.

Remark 2.6.6 For additional analogous properties for concave monotone functions the reader is directed to the book of Tuy (1998).

Summary

In this chapter, the basic elements of convex analysis are introduced. Section 2.1 presents the definitions and properties of convex sets, the definitions of convex combination and convex hull. Section 2.2 discusses the definitions and properties of convex and concave functions, the definitions of continuity, lower and upper semi-continuity of functions, the definitions and properties of differentiable convex and concave functions, and the conditions of convexity and concavity along with their associated tests. Section 2.3 focuses on the generalizations of convex and concave functions and treats the quasi-convex, quasi-concave, pseudo-convex and pseudo-concave functions and their properties. Section 2.4 discusses the definition of the convex envelope along with its form for concave functions over a polytope and a simplex, as well as properties of the convex envelopes that include concave univariate functions, bilinear functions, products of univariate functions, and general univariate functions. Section 2.5 discusses the definitions of the local, global and ϵ-global minima. Finally, section 2.6 presents the definitions of the d.c. and locally d.c. functions, describes their properties, and outlines approaches for their representation.

I Biconvex and Bilinear Problems

3 THE GOP PRIMAL - RELAXED DUAL DECOMPOSITION APPROACH : THEORY

This chapter presents the theory of the GOP algorithm introduced by Floudas and Visweswaran (1990, 1993). The GOP algorithm follows the basic ideas of decomposition first developed by Benders (1962) for mixed integer linear programming, and later extended by Geoffrion (1972) for nonlinear programming. The decomposition and duality principles are presented in detail in Floudas (1995). In section 3.1, the problem is stated and the classes of mathematical models that meet the imposed conditions are identified. In section 3.2, the reader is introduced to the basics of duality theory. In section 3.3, the theoretical properties that form the basis for the GOP are presented. Section 3.4 discusses the treatment of infeasible primal problems. Section 3.5 presents the algorithmic steps of the GOP approach and section 3.6 focuses on the convergence to an ϵ-global optimum solution. Section 3.7 presents an illustrative example for the geometrical interpretation of the GOP algorithm. In section 3.8, the theoretical development of the GOP for quadratic programming problems with linear constraints is presented. In section 3.9, the GOP theory for quadratically constrained problems is discussed. In section 3.10, the theoretical analysis of the GOP for univariate polynomial problems is presented. Section 3.11 presents additional properties of the GOP that exploit the mathematical structure of the Lagrange function and enhance the computational performance. Section 3.12 presents the GOP approach in a branch and bound framework and its illustration. Section 3.13 introduces the reformulation of the

C. A. Floudas, *Deterministic Global Optimization*
© Springer Science+Business Media Dordrecht 2000

relaxed dual problems as a single MILP model. Finally, section 3.15 presents a linear branching scheme for the GOP algorithm.

3.1 PROBLEM STATEMENT

The problems addressed in this chapter take the form :

$$\begin{aligned} \min_{\boldsymbol{x},\boldsymbol{y}} \quad & f(\boldsymbol{x},\boldsymbol{y}) \\ s.t. \quad & \boldsymbol{g}(\boldsymbol{x},\boldsymbol{y}) \leq \boldsymbol{0} \\ & \boldsymbol{h}(\boldsymbol{x},\boldsymbol{y}) = \boldsymbol{0} \\ & \boldsymbol{x} \in X, \quad \boldsymbol{y} \in Y \end{aligned} \tag{3.1}$$

where X, Y are compact convex sets, and $\boldsymbol{g}(\boldsymbol{x},\boldsymbol{y})$ and $\boldsymbol{h}(\boldsymbol{x},\boldsymbol{y})$ are vectors of inequality and equality constraints. All the functions are given in analytical form and are differentiable. The following conditions, denoted as *Conditions (A)* need to be satisfied :

(a) $f(\boldsymbol{x},\boldsymbol{y})$ *is convex in* $\boldsymbol{x}$ *for every fixed* $\boldsymbol{y}$*, and convex in* $\boldsymbol{y}$ *for every fixed* $\boldsymbol{x}$*.*

(b) $\boldsymbol{g}(\boldsymbol{x},\boldsymbol{y})$ *are convex in* $\boldsymbol{x}$ *for every fixed* $\boldsymbol{y}$*, and convex in* $\boldsymbol{y}$ *for every fixed* $\boldsymbol{x}$*.*

(c) $\boldsymbol{h}(\boldsymbol{x},\boldsymbol{y})$ *are affine in* $\boldsymbol{x}$ *for every fixed* $\boldsymbol{y}$*, and affine in* $\boldsymbol{y}$ *for every fixed* $\boldsymbol{x}$*.*

(d) *An appropriate first order constraint qualification (e.g., Slater's constraint qualification) is satisfied for fixed* $\boldsymbol{y}$*.*

Given the aforementioned problem statement, we would like to identify the classes of mathematical problems that can be represented within the framework of (3.1) and satisfy *Conditions (A)*. To address this issue, the concepts of *partitioning* and *transformations* are introduced.

3.1.1 Partitioning of the variable set

Simple *partitioning* of the variable set is sufficient to ensure that (a), (b), and (c) of *Conditions (A)* are satisfied in the case of quadratic programming problems where the nonconvexities are due to the presence of bilinear terms in the objective function and/or constraints. For such problems, the *projection* of the problem into the space of a subset of the variables results in a convex programming problem. Hence, it is possible to *partition* the variable set into two subsets x and y in such a way that *Conditions (A)* are satisfied (Floudas *et al.*, 1989). This is illustrated by the following example, which will also be used in the following section.

Illustration 3.1.1

$$\begin{aligned} \min_{x} \quad & -x_1 - x_2 \\ s.t. \quad & x_1 x_2 \leq 4, \\ & 0 \leq x_1 \leq 4, \\ & 0 \leq x_2 \leq 8 . \end{aligned}$$

The nonconvexity in this problem arises due to the presence of the bilinear term $x_1 x_2$ in the first constraint. This naturally suggests *partitioning* the variable set into the subsets x_1 and x_2. Redefining x_1 as x and x_2 as y, it can be seen that for a fixed value of x (say x^k), the constraint is linear in y, and vice-versa. Hence, such a *partition* ensures that *Conditions (A)* are satisfied for this problem.

3.1.2 Partitioning and Transformations

Simple *partitioning* of the variables will not ensure that (a), (b), and (c) of *Conditions (A)* are satisfied when the nonconvexities are due to general quadratic or polynomial terms. This difficulty can be addressed by introducing new *transformation* variables and reformulating the problem so that the nonconvexities are due to bilinear terms in the objective function and/or constraint set.

Illustration 3.1.2 Consider the minimization of the following sixth order polynomial function in one variable subject to bound constraints (Wingo, 1985):

$$\min_{y} \ \frac{1}{6}y^6 - \frac{52}{25}y^5 + \frac{39}{80}y^4 + \frac{71}{10}y^3 - \frac{79}{20}y^2 - y + \frac{1}{10}$$

$$-2 \leq y \leq 11.$$

The nonconvexities are due to the terms containing $-y^2$, y^3 and $-y^5$. By introducing five new variables $x_1, x_2, \ldots, x_5$, and adding the constraints

$$x_1 - y = 0, \ x_2 - x_1 y = 0, \ x_3 - x_2 y = 0, \ x_4 - x_3 y = 0, \ \text{ and } \ x_5 - x_4 y = 0,$$

the problem can be converted to a form where the variable set can be partitioned, as shown below:

$$\begin{aligned} \min \quad & \frac{1}{6}y^6 - \frac{52}{25}x_5 + \frac{39}{80}x_4 + \frac{71}{10}x_3 - \frac{79}{20}x_2 - x_1 + \frac{1}{10} \\ s.t. \quad & x_1 - y = 0, \\ & x_2 - x_1 y = 0, \\ & x_3 - x_2 y = 0, \\ & x_4 - x_3 y = 0, \\ & x_5 - x_4 y = 0, \\ & -2 \leq y \leq 11, \\ & x_i^L \leq x_i \leq x_i^U \end{aligned}$$

$$\boldsymbol{x}^L = (-2, 0, -8, 0, -32), \qquad \boldsymbol{x}^U = (11, 121, 1331, 14641, 161051).$$

For every fixed $y = y^k$, the objective function and the equality constraints are linear in x_i, and for every fixed x_i, the objective function is convex in y, and the equality constraints are linear in y. Hence, the reformulated problem satisfies (a), (b), and (c) of *Conditions (A)*.

Such a reformulation is possible for (a) quadratic programming problems with linear and/or quadratic constraints (including pooling/blending problems), and (b) certain classes of optimization problems involving polynomial functions of one or more variables in the objective function and/or the constraint set.

For larger problems, the choice of subsets for *partitioning* is not apparent. Floudas *et al.* (1989) showed that graph theory can be used to determine the partitions that satisfy (a), (b), and (c) of *Conditions (A)*. Hansen and Jaumard (1992) proposed graph theoretical results and an algorithm for the efficient bilinearization of quadratic and polynomial function problems, rational polynomials, and problems involving hyperbolic functions. This approach provides the set of new variables that convert the problem into bilinear form. The bilinearization can be achieved with the objective of minimizing either the number of $\boldsymbol{x}$ (or $\boldsymbol{y}$) variables or the number of new variables that make the problem completely bilinear. Moreover, given a bilinear problem with variable subsets $\boldsymbol{x}$ and $\boldsymbol{y}$, the algorithm can also be used to identify any changes in these subsets that will result in a smaller number of either the $\boldsymbol{x}$ or the $\boldsymbol{y}$ variables.

3.2 DUALITY THEORY

In this section, we present the basics of duality theory. For a detailed presentation, the reader is directed to the textbook of Floudas (1995). Since problem (3.1) satisfies *Conditions (A)*, fixing the $\boldsymbol{y}$ variables (say to $\boldsymbol{y}^k \in Y$) results in a subproblem that is much easier to solve:

$$\begin{aligned} \min \quad & f(\boldsymbol{x}, \boldsymbol{y}^k) \\ s.t. \quad & \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{y}^k) \leq \boldsymbol{0} \\ & \boldsymbol{h}(\boldsymbol{x}, \boldsymbol{y}^k) = \boldsymbol{0}. \end{aligned} \tag{3.2}$$

This problem is called the *Primal Problem*. For the sake of convenience, it is assumed here that any bounds on the $\boldsymbol{x}$ variables (as given by the compact set X) are incorporated into the first set of constraints. Due to the restriction of the $\boldsymbol{y}$ variables, this is equivalent to solving the original problem with some additional constraints, and therefore the solution of problem (3.2) (if feasible) provides an upper bound on the solution of problem (3.1). Moreover, since this problem is convex (by *Conditions (A)*) it can be solved by any conventional nonlinear *local* optimization solver. The solution of this problem also provides a set of multipliers (λ^k, μ^k) for the equality and inequality constraints respectively.

Illustration 3.2.1 We consider here the illustrative example 3.1.1. Suppose that $y^1 = 2$. Then, the primal problem is given by

$$\begin{aligned} \min_{x,y} \; & -x - 2 \\ s.t. \quad & 2x \leq 4 \\ & 0 - x \leq 0 \\ & x - 4 \leq 0. \end{aligned}$$

It should be noted that the bounds on x have been added as explicit constraints. The solution of this problem is $x = 2$, $f = -4$, and thus provides an upper bound on the optimal solution of $f^\star = -8.5$. The multiplier for the first constraint is $\mu_1^1 = 0.5$, and $\mu_2^1 = \mu_3^1 = 0$.

Problem (3.1) is equivalent to the minimization over the $\boldsymbol{x}$ and $\boldsymbol{y}$ variables occurs separately:

$$\begin{aligned} \min_{\boldsymbol{x}} \; \min_{\boldsymbol{y}} \; & f(\boldsymbol{x}, \boldsymbol{y}) \\ s.t. \quad & \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{y}) \leq \boldsymbol{0} \\ & \boldsymbol{h}(\boldsymbol{x}, \boldsymbol{y}) = \boldsymbol{0} \\ & \boldsymbol{x} \in X, \quad \boldsymbol{y} \in Y. \end{aligned}$$

The *projection* of this problem in the space of the $\boldsymbol{y}$ variables (Geoffrion, 1972) further enables (3.1) to be written in the following form, featuring inner and outer optimization problems:

$$\begin{aligned} & \min_{\boldsymbol{y}} \quad v(\boldsymbol{y}) \\ s.t. \quad & v(\boldsymbol{y}) = \min_{\boldsymbol{x} \in X} f(\boldsymbol{x}, \boldsymbol{y}) \\ & \qquad s.t. \quad \boldsymbol{h}(\boldsymbol{x}, \boldsymbol{y}) = \boldsymbol{0} \\ & \qquad\quad\;\; \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{y}) \leq \boldsymbol{0} \\ & \boldsymbol{y} \in Y \cap V \\ where \quad & V \equiv \{\boldsymbol{y} \; : \; \boldsymbol{h}(\boldsymbol{x}, \boldsymbol{y}) = \boldsymbol{0}, \; \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{y}) \leq \boldsymbol{0} \;\text{ for some } \boldsymbol{x} \in X\}. \end{aligned} \tag{3.3}$$

In formulation (3.3), the inner minimization problem (in the first constraint) is parametric in $\boldsymbol{y}$. For any fixed value of $\boldsymbol{y}$, say $\boldsymbol{y}^k$, this problem is simply the primal problem (3.2) solved for that $\boldsymbol{y} = \boldsymbol{y}^k$. Therefore, the function $v(\boldsymbol{y})$ can be defined as the set of solutions of (3.2) for different values of $\boldsymbol{y}$.

It is important to note that $v(\boldsymbol{y})$ is known only implicitly. For this reason, problem (3.3) can be as difficult to solve as the original problem. To simplify this problem, we turn to the dual form of (3.1).

From the *Strong Duality Theorem* (Bazaraa *et al.*, 1993; Floudas, 1995), the solution of (3.2), for any fixed $\boldsymbol{y} = \boldsymbol{y}^k$, is identical to the solution of its corresponding dual problem. That is,

$$\left\{\begin{array}{lll} \min\limits_{\boldsymbol{x}\in X} f(\boldsymbol{x},\boldsymbol{y}^k) & & \\ s.t. \quad \boldsymbol{g}(\boldsymbol{x},\boldsymbol{y}^k) & \leq & \boldsymbol{0} \\ \quad\quad \boldsymbol{h}(\boldsymbol{x},\boldsymbol{y}^k) & = & \boldsymbol{0} \end{array}\right\} = \sup_{\substack{\mu\geq 0\\ \lambda}} \inf_{\boldsymbol{x}\in X} \left\{f(\boldsymbol{x},\boldsymbol{y}^k) + \lambda^T\boldsymbol{h}(\boldsymbol{x},\boldsymbol{y}^k) + \mu^T\boldsymbol{g}(\boldsymbol{x},\boldsymbol{y}^k)\right\}$$

$$\text{forall} \quad \boldsymbol{y}^k \in Y \cap V.$$

where λ and μ are the vectors of Lagrange multipliers corresponding to the equality and inequality constraints of the primal problem (3.2).

Using the Strong Duality Theorem, the set $v(\boldsymbol{y})$ can be rewritten as

$$v(\boldsymbol{y}) = \sup_{\substack{\mu\geq 0\\ \lambda}} \inf_{\boldsymbol{x}\in X} \left\{f(\boldsymbol{x},\boldsymbol{y}) + \lambda^T\boldsymbol{h}(\boldsymbol{x},\boldsymbol{y}) + \mu^T\boldsymbol{g}(\boldsymbol{x},\boldsymbol{y})\right\} \quad \forall\, \boldsymbol{y} \in Y \cap V.$$

From the definition of supremum, the maximization over λ and μ can be relaxed to an upper bound:

$$v(\boldsymbol{y}) \geq \inf_{\boldsymbol{x}\in X} \left\{\, f(\boldsymbol{x},\boldsymbol{y}) + \lambda^T\boldsymbol{h}(\boldsymbol{x},\boldsymbol{y}) + \mu^T\boldsymbol{g}(\boldsymbol{x},\boldsymbol{y}) \,\right\} \quad \text{forall} \quad \mu \geq 0, \lambda.$$

Thus, assuming the existence of feasible solutions to the inner minimization problems, the dual representation of $v(\boldsymbol{y})$ leads to the following formulation, equivalent to (3.3):

$$\begin{array}{rcl} & & \min\limits_{\boldsymbol{y}} \; v(\boldsymbol{y}) \\ s.t. & & \\ v(\boldsymbol{y}) & \geq & \min\limits_{\boldsymbol{x}\in X} \left\{\, f(\boldsymbol{x},\boldsymbol{y}) + \lambda^T\boldsymbol{h}(\boldsymbol{x},\boldsymbol{y}) + \mu^T\boldsymbol{g}(\boldsymbol{x},\boldsymbol{y}) \,\right\}, \quad \forall \mu \geq 0, \lambda \quad (3.4) \\ \boldsymbol{y} & \in & Y \cap V \\ V & \equiv & \{\boldsymbol{y} : \boldsymbol{h}(\boldsymbol{x},\boldsymbol{y}) = \boldsymbol{0},\ \boldsymbol{g}(\boldsymbol{x},\boldsymbol{y}) \leq \boldsymbol{0} \ \text{ for some } \boldsymbol{x} \in X\}. \end{array}$$

The last two sets of constraints in problem (3.4) represent an implicitly defined set in $\boldsymbol{y}$. Therefore, the presence of these constraints makes the solution of (3.4) extremely difficult. This can be avoided by simply dropping these two sets of constraints from (3.4). There is one case where this is unnecessary. When the constraints for the original problem (3.1) are convex, they can be directly incorporated into the dual problem.

By dropping the last two constraints from (3.4), the *Relaxed Dual* problem is obtained:

$$\begin{array}{rcl} & & \min\limits_{\substack{\boldsymbol{y}\in Y\\ \mu_B}} \; \mu_B \\ s.t. & & \\ \mu_B & \geq & \min\limits_{\boldsymbol{x}\in X} L(\boldsymbol{x},\boldsymbol{y},\lambda^k,\mu^k) \qquad \forall \mu^k \geq 0, \lambda^k \end{array} \tag{3.5}$$

where μ_B is a scalar, and the function

$$L(\boldsymbol{x},\boldsymbol{y},\lambda^k,\mu^k) = f(\boldsymbol{x},\boldsymbol{y}) + {\lambda^k}^T\boldsymbol{h}(\boldsymbol{x},\boldsymbol{y}) + {\mu^k}^T\boldsymbol{g}(\boldsymbol{x},\boldsymbol{y}) \tag{3.6}$$

is the Lagrange function for problem (3.2). The multipliers μ^k and λ^k are obtained by solving a primal problem for $\boldsymbol{y} = \boldsymbol{y}^k$.

The inner minimization problem, that is, the problem of minimizing the Lagrange function with respect to the $\boldsymbol{x}$ variables, is called the *Inner Relaxed Dual* problem, and is given by:

$$L^\star(\overline{\boldsymbol{x}}, \boldsymbol{y}, \lambda^k, \mu^k) = \begin{cases} \min\limits_{\boldsymbol{x} \in X} \; L(\boldsymbol{x}, \boldsymbol{y}, \lambda^k, \mu^k) \\ L(\boldsymbol{x}, \boldsymbol{y}, \lambda^k, \mu^k) = f(\boldsymbol{x}, \boldsymbol{y}) + {\lambda^k}^T \boldsymbol{h}(\boldsymbol{x}, \boldsymbol{y}) + {\mu^k}^T \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{y}). \end{cases} \tag{3.7}$$

Here, $\overline{\boldsymbol{x}}$ denotes the value of the $\boldsymbol{x}$ variables at the solution of (3.7).

Illustration 3.2.2 For $y^1 = 2$, the solution to the primal problem (3.2) yields $\mu^1 = 0.5$. Thus, the Lagrange function can be formulated as

$$L(x, y, \mu^1) = -x - y + 0.5(xy - 4) = 0.5x(y - 2) - y - 2$$

and the relaxed dual problem becomes

$$\begin{aligned} & \min_{y, \mu_B} \; \mu_B \\ s.t. \quad \mu_B & \geq 0.5x(y-2) - y - 2 \\ 0 & \leq y \leq 8. \end{aligned}$$

The use of projection and duality theory as presented thus provides an obvious way of obtaining upper and lower bounds on the global solution of (3.1). The primal problem (3.2), represents an upper bound on the original problem (3.1). The relaxed dual problem in the form (3.5) contains fewer constraints than (3.4). Hence, it represents a valid lower bound on the original problem (3.1). Thus, these two problems can be used in an iterative fashion to determine a global solution of (3.1).

3.3 MATHEMATICAL PROPERTIES

In the form given by (3.5), the relaxed dual problem is difficult to solve, since it contains the inner relaxed dual problem, which is parametric in $\boldsymbol{y}$. Therefore, we need to determine an one-level optimization problem that provides a valid underestimator of the relaxed dual problem, enabling it to be solved rigorously. To determine such an one-level optimization problem, we utilize the following properties:

Property 3.3.1 *The solution of each primal subproblem of the form (3.2) is the global solution of that problem.*

Proof. From the criteria given by *Conditions (A)*, the primal problem is a convex programming problem in $\boldsymbol{x}$. Hence, the solution of every primal is its global solution (Avriel, 1976).

Illustration 3.3.1 For a fixed $\boldsymbol{y}$, the primal problem is a linear programming problem in $\boldsymbol{x}$. Therefore, it is convex, and this property holds.

Property 3.3.2 *The Lagrange function of the kth iteration, $L(\boldsymbol{x}, \boldsymbol{y}, \lambda^k, \mu^k)$, satisfies the following Conditions (B) :*

(i) *$L(\boldsymbol{x}, \boldsymbol{y}, \lambda^k, \mu^k)$ is convex in $\boldsymbol{x}$ $\forall$ $\boldsymbol{y} = \boldsymbol{y}^d$, and*

(ii) *$L(\boldsymbol{x}, \boldsymbol{y}, \lambda^k, \mu^k)$ is convex in $\boldsymbol{y}$ $\forall$ $\boldsymbol{x} = \boldsymbol{x}^k$.*

Proof. The Lagrange function is a weighted sum of the objective function, equality and inequality constraints of the primal subproblem. Therefore, for each discretized $\boldsymbol{y} = \boldsymbol{y}^d$ (in this context, discretized implies $\boldsymbol{y}$ is kept fixed at $\boldsymbol{y}^d$) the Lagrange function is convex in $\boldsymbol{x}$ since it is a sum of convex functions in $\boldsymbol{x}$ (by *Conditions (A)*). Similarly, for every fixed $\boldsymbol{x} = \boldsymbol{x}^k$, the Lagrange function is a sum of convex functions in $\boldsymbol{y}$.

Illustration 3.3.2 For this example,

$$L(x, y, \mu^k) = -x - y + \mu_1^k(xy - 4) + \mu_2^k(0 - x) + \mu_3^k(x - 4).$$

As can be seen, once either x or y is fixed, the Lagrange function is linear in the other variable.

Property 3.3.3 *The solution of the* inner relaxed dual *problem (3.7) is its global solution for each discretized $\boldsymbol{y} = \boldsymbol{y}^d$.*

Proof. From Property 3.3.2, the Lagrange function is convex in $\boldsymbol{x}$ for every fixed (discretized) $\boldsymbol{y} = \boldsymbol{y}^d$. Therefore, the inner relaxed dual problem also meets the global optimality requirements (Avriel, 1976).

Property 3.3.4

$$\min_{\boldsymbol{x} \in X} L(\boldsymbol{x}, \boldsymbol{y}^d, \lambda^k, \mu^k) \geq \min_{\boldsymbol{x} \in X} L(\boldsymbol{x}, \boldsymbol{y}^d, \lambda^k, \mu^k)\Big|_{\boldsymbol{x}^k}^{lin} \qquad \forall \boldsymbol{y} = \boldsymbol{y}^d \tag{3.8}$$

where $L(\boldsymbol{x}, \boldsymbol{y}^d, \lambda^k, \mu^k)|_{\boldsymbol{x}^k}^{lin}$ is the linearization of the Lagrange function $L(\boldsymbol{x}, \boldsymbol{y}^d, \lambda^k, \mu^k)$ around $\boldsymbol{x}^k$, the solution of the kth primal subproblem.

Proof. For every discretized $\boldsymbol{y} = \boldsymbol{y}^d$, the Lagrange function $L(\boldsymbol{x}, \boldsymbol{y}^d, \lambda^k, \mu^k)$ is convex in $\boldsymbol{x}$. Hence, the linearization of the Lagrange function around some $\boldsymbol{x}$ is a valid underestimator of the function for all values of $\boldsymbol{x}$. This is true regardless of the point of linearization of $\boldsymbol{x}$,

$$L(\boldsymbol{x}, \boldsymbol{y}^d, \lambda^k, \mu^k) \geq L(\boldsymbol{x}, \boldsymbol{y}^d, \lambda^k, \mu^k)\Big|_{\boldsymbol{x}^k}^{lin} \qquad \forall \boldsymbol{y} = \boldsymbol{y}^d. \tag{3.9}$$

Since this applies for every $\boldsymbol{x}$, the minimum over $\boldsymbol{x}$ can be taken on both sides leading to the desired result.

Illustration 3.3.3 The Lagrange function for the example is bilinear. Therefore, the linearization with respect to x yields the same function, and the property holds as an equality.

The important definition of *Connected Variables* is introduced next.

Definition 3.3.1 (Connected Variables) *Define the vector $\boldsymbol{g}^k(\boldsymbol{y})$ (consisting of elements $g_i^k(\boldsymbol{y})$) as follows :*

$$\boldsymbol{g}^k(\boldsymbol{y}) \;=\; \nabla_{\boldsymbol{x}} L(\boldsymbol{x}, \boldsymbol{y}, \lambda^k, \mu^k)\big|_{\boldsymbol{x}^k}\,, \quad \text{and} \quad g_i^k(\boldsymbol{y}) \;=\; \nabla_{x_i} L(\boldsymbol{x}, \boldsymbol{y}, \lambda^k, \mu^k)\big|_{x^k}\,,$$

where x_i is the ith element of the vector $\boldsymbol{x}$, $i = 1, 2, \ldots, n$. Then, it can be seen that for every fixed $\boldsymbol{y}$, the linearization of the Lagrange function at $\boldsymbol{x}^k$ is given by

$$\begin{aligned} L(\boldsymbol{x}, \boldsymbol{y}, \lambda^k, \mu^k)\big|_{\boldsymbol{x}^k}^{lin} &= L(\boldsymbol{x}^k, \boldsymbol{y}, \lambda^k, \mu^k) + \boldsymbol{g}^k(\boldsymbol{y})^T \cdot (\boldsymbol{x} - \boldsymbol{x}^k) \\ &= L(\boldsymbol{x}^k, \boldsymbol{y}, \lambda^k, \mu^k) + \sum_{i=1}^{n} g_i^k(\boldsymbol{y}) \cdot (x_i - x_i^k). \end{aligned}$$

The Karush-Kuhn-Tucker (KKT) optimality conditions for the primal problem require that

$$g_i^k(\boldsymbol{y}^k) = \nabla_{x_i} L(\boldsymbol{x}, \boldsymbol{y}^k, \lambda^k, \mu^k)\big|_{\boldsymbol{x}=\boldsymbol{x}^k} = 0 \qquad \forall\, i = 1, \ldots, n. \tag{3.10}$$

Then, for $\boldsymbol{y} = \boldsymbol{y}^k$, every variable x_i for which $g_i^k(\boldsymbol{y})$ is a function of $\boldsymbol{y}$ is defined to be a connected *variable. The set of all such* connected *variables is given by I_c^k, that is,*

$$I_c^k \equiv \{i : g_i^k(\boldsymbol{y}) \text{ is a function of } \boldsymbol{y}\}\ .$$

Illustration 3.3.4 For the example, the Lagrange function can be rewritten as

$$L(x, y, \mu^k) = -x - y + \mu_1^k(xy - 4) + \mu_2^k(0 - x) + \mu_3^k(x - 4).$$

Therefore, the KKT conditions are given by

$$g^k(y^k) = \nabla_x L(x, y^k, \mu^k)\big|_{x=x^k} = \mu_1^k y^k - \mu_2^k + \mu_3^k - 1 = 0.$$

Since the coefficient of x in the objective is negative, the solution of the primal problem for any $y = y^k$ must satisfy either the constraint $xy^k \leq 4$ or $x \leq 4$ as a strict equality. More specifically:

(i) If $y^k < 1$, then $x = 4, \mu_3^k = 1, \mu_1^k = \mu_2^k = 0$, and

$$L(x, y, \mu^k) = -x - y + 1(x - 4) = -y - 4.$$

In this case, $g^k(y) = 0$, and the set I_c^k is empty.

(ii) If $y^k \geq 1$, then $x = 4/y^k$, $\mu_1^k = 1/y^k$, $\mu_2^k = \mu_3^k = 0$, and

$$\begin{aligned} L(x, y, \mu^k) &= -x - y + \frac{1}{y^k}(xy - 4) \\ &= -x^k - y + \mu_1^k x^k y - 4\mu_1^k + \mu_1^k (y - y^k)(x - x^k). \end{aligned}$$

Thus, $g^k(y) = \mu_1^k (y - y^k)$, and the set I_c^k consists of x.

Property 3.3.5 *The optimal solution of the inner relaxed dual (3.7), with the Lagrange function replaced by its linearization at $\boldsymbol{x}^k$, depends only on those x_i, for which $g_i^k(\boldsymbol{y})$ is a function of $\boldsymbol{y}$ (i.e., the* connected $\boldsymbol{x}$ *variables.)*

Proof. The linearization of the Lagrange function can be written as

$$L(\boldsymbol{x}, \boldsymbol{y}, \lambda^k, \mu^k)\big|_{\boldsymbol{x}^k}^{lin} = L(\boldsymbol{x}^k, \boldsymbol{y}, \lambda^k, \mu^k) + \sum_{\substack{i=1 \\ i \neq j}}^{n} g_i^k(\boldsymbol{y}) \cdot (x_i - x_i^k) + g_j^k(\boldsymbol{y}) \cdot (x_j - x_j^k). \tag{3.11}$$

Using Property 3.3.2 and the KKT conditions (3.10), the inner relaxed dual problem can be written as

$$\begin{aligned} L^*(\overline{\boldsymbol{x}}, \boldsymbol{y}, \lambda^k, \mu^k) &\geq \min_{\boldsymbol{x}} L(\boldsymbol{x}, \boldsymbol{y}, \lambda^k, \mu^k)\big|_{\boldsymbol{x}^k}^{lin} \\ &\geq L(\boldsymbol{x}^k, \boldsymbol{y}, \lambda^k, \mu^k) \\ &+ \min_{\boldsymbol{x}} \left\{ \sum_{\substack{i=1 \\ i \neq j}}^{n} g_i^k(\boldsymbol{y}) \cdot (x_i - x_i^k) + g_j^k(\boldsymbol{y})(x_j - x_j^k) \right\}. \end{aligned}$$

Now, suppose that $g_j^k(\boldsymbol{y})$ is not a function of $\boldsymbol{y}$. Then, by (3.10), $g_j^k(\boldsymbol{y}) = g_j^k(\boldsymbol{y}^k) = 0$. Therefore,

$$L^*(\overline{\boldsymbol{x}}, \boldsymbol{y}, \lambda^k, \mu^k) \geq L(\boldsymbol{x}^k, \boldsymbol{y}, \lambda^k, \mu^k) + \min_{\boldsymbol{x}} \{ \sum_{\substack{i=1 \\ i \neq j}}^{n} g_i^k(\boldsymbol{y}) \cdot (x_i - x_i^k) \}.$$

Hence, the linearization of the Lagrange function does not depend on x_j, and the minimization of the Lagrange function in its linearized form with respect to x_j will not have any effect on the solution of the inner relaxed dual problem.

Remark 3.3.1 This property is important from the computational point of view. It implies that the inner relaxed dual can be replaced by a problem involving the minimization of the linearized Lagrange function over the set of

connected $\boldsymbol{x}$ variables. For a number of optimization problems, the nonconvexities are limited to a small number of terms and variables. This property suggests that for such problems, the computational effort required to obtain a global solution is not determined by the number of variables in the problem, but rather by the number of *connected* variables, which can result in reductions by several orders of magnitude in the time needed to solve the problem.

Illustration 3.3.5 Suppose that $y^k = 0$. Then, the solution of the primal problem is $x^k = 4$, $\mu_3^k = 1$, $\mu_1^k = \mu_2^k = 0$, and

$$L(x, y, \mu^k) = -x - y + 1(x - 4) = -y - 4.$$

It can be seen that the solution of the inner relaxed dual problem does not depend on x.

Property 3.3.6 *For any given* $\boldsymbol{y}^k$ *and for all* $i \in I_c^k$, *let* x_i^L *and* x_i^U *be the lower and upper bounds on the variable* x_i. *Also, let* B_j *be a combination of lower/upper bounds of these* connected $\boldsymbol{x}$ *variables and let* CB *be the set of all such combinations. Then,*

$$L^*(\overline{\boldsymbol{x}}, \boldsymbol{y}, \lambda^k, \mu^k) \geq \min_{B_j \in CB} \left\{ \begin{array}{ll} & L(\boldsymbol{x}^{B_j}, y, \lambda^k, \mu^k)\big|_{\boldsymbol{x}^k}^{lin}, \\ \textit{with} & g_i^k(\boldsymbol{y}) \geq 0 \quad \forall x_i^{B_j} = x_i^L, \\ & g_i^k(\boldsymbol{y}) \leq 0 \quad \forall x_i^{B_j} = x_i^U \end{array} \right\} \quad \forall\, \boldsymbol{y}. \quad (3.12)$$

Proof. From Property 3.3.4,

$$L^*(\overline{\boldsymbol{x}}, \boldsymbol{y}, \lambda^k, \mu^k) \geq \min_{\boldsymbol{x}} \; L(\boldsymbol{x}, \boldsymbol{y}, \lambda^k, \mu^k)\big|_{\boldsymbol{x}^k}^{lin} \quad \forall\, \boldsymbol{y}. \quad (3.13)$$

Using the definition of $L(\boldsymbol{x}, \boldsymbol{y}, \lambda^k, \mu^k)\big|_{\boldsymbol{x}^k}^{lin}$, the right hand side of (3.13) is given by

$$\begin{aligned} \min_{\boldsymbol{x}} \; L(\boldsymbol{x}, \boldsymbol{y}, \lambda^k, \mu^k)\big|_{\boldsymbol{x}^k}^{lin} &= \min_{\boldsymbol{x}} \Big[L(\boldsymbol{x}^k, \boldsymbol{y}, \lambda^k, \mu^k) + \sum_{i \in I_c^k} g_i^k(\boldsymbol{y})(x_i - x_i^k) \Big] \\ &= L(\boldsymbol{x}^k, \boldsymbol{y}, \lambda^k, \mu^k) + \min_{\boldsymbol{x}} \sum_{i \in I_c^k} g_i^k(\boldsymbol{y})(x_i - x_i^k). \end{aligned}$$

For any fixed $\boldsymbol{y} = \boldsymbol{y}^d$, the operators for minimization and summation can be exchanged. Hence,

$$\min_{\boldsymbol{x}} \; L(\boldsymbol{x}, \boldsymbol{y}, \lambda^k, \mu^k)\big|_{\boldsymbol{x}^k}^{lin} = L(\boldsymbol{x}^k, \boldsymbol{y}, \lambda^k, \mu^k) + \sum_{i \in I_c^k} \min_{x_i} \; g_i^k(\boldsymbol{y})(x_i - x_i^k). \quad (3.14)$$

Consider the ith component of the second term on the right hand side. It is linear in x_i. Hence, the minimum of this term will lie at a bound of x_i, the

specific nature of the bound (lower or upper) being determined by the sign of $\nabla_{x_i} L(\boldsymbol{x}, \boldsymbol{y}, \lambda^k, \mu^k)\big|_{\boldsymbol{x}^k}$. Two cases are possible:

(a) If $g_i^k(\boldsymbol{y}) \geq 0$, then

$$\min_{x_i} \; g_i^k(\boldsymbol{y})(x_i - x_i^k) \geq g_i^k(\boldsymbol{y})(x_i^L - x_i^k) \; .$$

(b) If $g_i^k(\boldsymbol{y}) \leq 0$, then

$$\min_{x_i} g_i^k(\boldsymbol{y})(x_i - x_i^k) \geq g_i^k(\boldsymbol{y})(x_i^U - x_i^k) \; .$$

These two cases can be combined to yield the following result:

$$\min_{x_i} g_i^k(\boldsymbol{y})(x_i - x_i^k) \geq g_i^k(\boldsymbol{y})(x_i^{B_j} - x_i^k),$$

where (3.15)

$$x_i^{B_j} = \begin{cases} x_i^L & \forall\, \boldsymbol{y} : \; g_i^k(\boldsymbol{y}) \geq 0 \\ x_i^U & \forall\, \boldsymbol{y} : \; g_i^k(\boldsymbol{y}) \leq 0 \end{cases}$$

Combining (3.13), (3.14) and (3.15), it can be seen that

$$L^*(\overline{\boldsymbol{x}}, \boldsymbol{y}, \lambda^k, \mu^k) \;\; \geq \;\; L(\boldsymbol{x}^k, \boldsymbol{y}, \lambda^k, \mu^k) + \sum_{i \in I_c^k} g_i^k(\boldsymbol{y})(x_i^{B_j} - x_i^k), \qquad (3.16)$$

where

$$\begin{aligned} x_i^{B_j} &= x_i^L \quad \forall \boldsymbol{y} : \; g_i^k(\boldsymbol{y}) \geq 0 \\ x_i^{B_j} &= x_i^U \quad \forall \boldsymbol{y} : \; g_i^k(\boldsymbol{y}) \leq 0. \end{aligned}$$

From this, it is evident that for any fixed $\boldsymbol{y} = \boldsymbol{y}^d$, there exists a combination of bounds B_j for the *connected* $\boldsymbol{x}$ variables such that

$$\begin{aligned} \min_{\boldsymbol{x}} L(\boldsymbol{x}, \boldsymbol{y}^d, \lambda^k, \mu^k) \;\; &\geq \;\; L(\boldsymbol{x}^k, \boldsymbol{y}^d, \lambda^k, \mu^k) + \sum_{i \in I_c^k} g_i^k(\boldsymbol{y})(x_i^{B_j} - x_i^k) \\ &\geq \;\; L(\boldsymbol{x}^{B_j}, \boldsymbol{y}^d, \lambda^k, \mu^k)\big|_{\boldsymbol{x}^k}^{lin}. \end{aligned}$$

Hence, for every discretized $\boldsymbol{y} = \boldsymbol{y}^d$, by fixing the values of the $\boldsymbol{x}$ variables at a combination of bounds B_j in the linearized Lagrange function and taking the minimum over all possible combinations of bounds $B_j \in CB$, a lower bound on the value of $L^*(\overline{\boldsymbol{x}}, \boldsymbol{y}^d, \lambda^k, \mu^k)$ is obtained. Since this is true for every $\boldsymbol{y} = \boldsymbol{y}^d$, (3.12) must hold for all $\boldsymbol{y}$.

Illustration 3.3.6 For $y^1 = 2$, the Lagrange function and its gradient are given by

$$\begin{aligned} L(x, y, \mu^1) \;\; &= \;\; 0.5x(y-2) - y - 2, \qquad \text{and} \\ g^1(y, \mu^1) \;\; &= \;\; 0.5(y-2). \end{aligned}$$

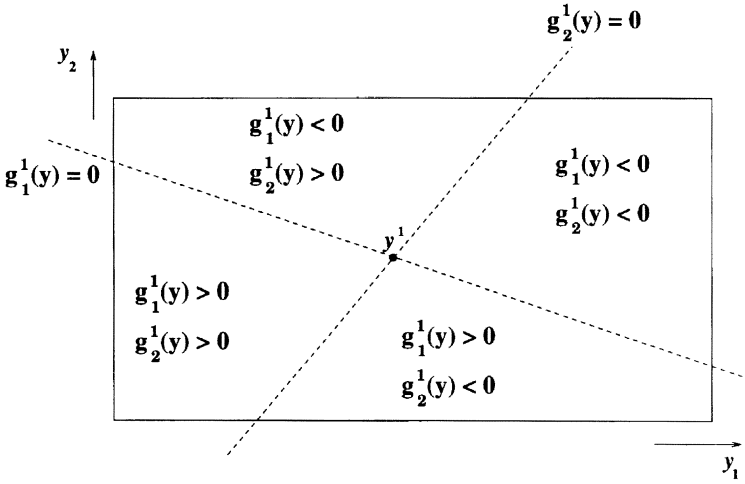


Figure 3.1. Illustration of Property 3.3.6

Consider the following two functions $U_1(y)$ and $U_2(y)$, obtained by substituting $x = 0$ and $x = 4$ (lower and upper bounds on x) respectively into $L(x, y, \mu^1)$:

$$\begin{aligned} U_1(y) &= -y - 2 \\ U_2(y) &= y - 6. \end{aligned}$$

It can then be seen that since $0 \leq x \leq 4$,

$$\begin{aligned} L(x, y, \mu^1) - U_1(y) &= 0.5x(y-2) > 0 \quad \forall\, y > 2 \\ L(x, y, \mu^1) - U_2(y) &= 0.5(x-4)(y-2) > 0 \quad \forall\, y < 2. \end{aligned}$$

Thus, for any value of y, there is a function (either $U_1(y)$ or $U_2(y)$) that underestimates the Lagrange function. Hence, for any value of y, $L^\star(y)$ is underestimated by the minimum of $U_1(y)$ and $U_2(y)$.

Figure 3.1 shows a hypothetical example with two x and two y variables. The gradients $g_1^1(\boldsymbol{y})$ and $g_2^1(\boldsymbol{y})$ divide the Y space into four regions. In each region, the Lagrange function will be underestimated by substituting an appropriate set of bounds for the $\boldsymbol{x}$ variables. Property 3.3.6 implies that it is sufficient to set the *connected* variables x_i , $i \in I_c$ to a set of bounds B_j, and solving the relaxed dual problem once for every such combination of bounds.

As can be seen, a new set of constraints (the gradients of the Lagrange function) have been introduced into (3.12). These constraints are called the *qualifying* constraints of the Lagrange function and might be introduced either as $g_i^k(\boldsymbol{y}) \geq 0$ or $g_i^k(\boldsymbol{y}) \leq 0$; to ensure convexity in the relaxed dual problem, this requires that $g_i^k(\boldsymbol{y})$ are linear in $\boldsymbol{y}$ for all i. Even in the general case of

biconvex functions (i.e., functions which satisfy *Conditions (A)* but are not necessarily bilinear, quadratic or polynomial) it is easy to satisfy this property by simply linearizing the Lagrange function with respect to $\boldsymbol{y}$. Because we seek a lower bound by minimizing the Lagrange function (which is convex in $\boldsymbol{y}$) the linearization is valid.

Property 3.3.7 *At the Kth iteration, define*

(i) *UL(k,K) to be the set of Lagrange functions from the kth iteration ($k < K$) whose qualifying constraints are satisfied at $\boldsymbol{y}^K$, the fixed value of the $\boldsymbol{y}$ variables for the primal problem at the Kth iteration.*

(ii) $\overline{\mu}_B{}^K$ *to be the optimal value of the Kth Relaxed Dual Problem. That is,*

$$\overline{\mu}_B{}^K = \left\{ \begin{array}{lll} & \min\limits_{\boldsymbol{y},\mu_B} & \mu_B \\ s.t. & \mu_B \geq \min\limits_{\boldsymbol{x}\in X} & L(\boldsymbol{x},\boldsymbol{y},\lambda^k,\mu^k) \quad k = 1,2\ldots,K-1 \\ & \mu_B \geq \min\limits_{\boldsymbol{x}\in X} & L(\boldsymbol{x},\boldsymbol{y},\lambda^K,\mu^K) \end{array} \right\}$$

Then,

$$\overline{\mu}_B^K \geq \min_{B_l \in CB} \left\{ \begin{array}{lll} & \min\limits_{\boldsymbol{y},\mu_B} & \mu_B \\ \mu_B \geq & L(\boldsymbol{x}^{B_j},\boldsymbol{y},\lambda^k,\mu^k)\big|_{\boldsymbol{x}^k}^{lin} & \\ & \left. \begin{array}{ll} g_i^k(\boldsymbol{y}) \leq 0 & x_i^{B_j} = x_i^U \\ g_i^k(\boldsymbol{y}) \geq 0 & x_i^{B_j} = x_i^L \end{array} \right\} & \begin{array}{l} \forall j \in UL(k,K) \\ k = 1,2,\ldots,K-1 \end{array} \\ \mu_B \geq & L(\boldsymbol{x}^{B_l},\boldsymbol{y},\lambda^K,\mu^K)\big|_{\boldsymbol{x}^k}^{lin} & \\ & \begin{array}{ll} g_i^K(\boldsymbol{y}) \leq 0 & if\ x_i^{B_l} = x_i^U \\ g_i^K(\boldsymbol{y}) \geq 0 & if\ x_i^{B_l} = x_i^L \end{array} & \end{array} \right\} \tag{3.17}$$

Proof. This property is proved by induction.

(1) For Iteration 1

For $k = 1$, from Property 3.3.6,

$$\min_{\boldsymbol{x}\in X} L(\boldsymbol{x},\boldsymbol{y},\lambda^1,\mu^1) \geq \min_{B_l \in CB} \left\{ \begin{array}{ll} \multicolumn{2}{c}{L(\boldsymbol{x}^{B_l},\boldsymbol{y},\lambda^1,\mu^1)} \\ g_i^1(\boldsymbol{y}) \leq 0 & if\ x_i^{B_l} = x_i^U \\ g_i^1(\boldsymbol{y}) \geq 0 & if\ x_i^{B_l} = x_i^L \end{array} \right\}, \forall \boldsymbol{y}.$$

Since this holds for all $\boldsymbol{y}$,

$$\min_{\boldsymbol{y}\in Y} \left\{ \min_{\boldsymbol{x}\in X} L(\boldsymbol{x},\boldsymbol{y},\lambda^1,\mu^1) \right\} \geq \min_{\boldsymbol{y}\in Y} \left\{ \min_{B_l \in CB} \left\{ \begin{array}{ll} \multicolumn{2}{c}{L(\boldsymbol{x}^{B_l},\boldsymbol{y},\lambda^1,\mu^1)} \\ g_i^1(\boldsymbol{y}) \leq 0 & if\ x_i^{B_l} = x_i^U \\ g_i^1(\boldsymbol{y}) \geq 0 & if\ x_i^{B_l} = x_i^L \end{array} \right\} \right\}.$$

The operators on the right hand side can be interchanged since $L(\boldsymbol{x}^{B_l}, \boldsymbol{y}, \lambda^1, \mu^1)$ depends only on $\boldsymbol{y}$. Therefore,

$$\min_{\boldsymbol{y}\in Y}\left\{\min_{\boldsymbol{x}\in X} L(\boldsymbol{x},\boldsymbol{y},\lambda^1,\mu^1)\right\} \geq \min_{B_l\in CB}\left\{\min_{\boldsymbol{y}\in Y}\left\{\begin{array}{ll} L(\boldsymbol{x}^{B_l},\boldsymbol{y},\lambda^1,\mu^1) & \\ g_i^1(\boldsymbol{y}) \leq 0 & if\ x_i^{B_l} = x_i^U \\ g_i^1(\boldsymbol{y}) \geq 0 & if\ x_i^{B_l} = x_i^L \end{array}\right\}\right\}$$

and equivalently,

$$\left\{\begin{array}{ll} \min\limits_{\substack{\boldsymbol{y}\in Y\\ \mu_B}} \mu_B & \\ s.t. \quad \mu_B \geq \min\limits_{\boldsymbol{x}\in X} L(\boldsymbol{x},\boldsymbol{y},\lambda^1,\mu^1) & \end{array}\right\} \geq \min_{B_l\in CB}\left\{\begin{array}{ll} \min\limits_{\boldsymbol{y}\in Y,\mu_B} \mu_B & \\ s.t. \quad \mu_B \geq L(\boldsymbol{x}^{B_l},\boldsymbol{y},\lambda^1,\mu^1) & \\ g_i^1(\boldsymbol{y}) \leq 0 & if\ x_i^{B_l} = x_i^U \\ g_i^1(\boldsymbol{y}) \geq 0 & if\ x_i^{B_l} = x_i^L \end{array}\right\}$$

Hence, for iteration 1, the property is proved.

(2) For Iteration 2

$UL(1,2)$ represents the set of Lagrange functions from the first iteration whose qualifying constraints are satisfied at $\boldsymbol{y}^2$. For the second iteration, the Lagrange functions from the first iteration that belong to $UL(1,2)$ already have the value of $\boldsymbol{x}$ set to the appropriate bound. Therefore, these Lagrange functions are functions only of $\boldsymbol{y}$.

Thus, we need to show that if

$$\overline{\mu}_B{}^2 = \left\{\begin{array}{l} \min\limits_{\boldsymbol{y},\mu_B} \mu_B \\ s.t. \\ \left.\begin{array}{ll} \mu_B \geq & L(\boldsymbol{x}^{B_j},\boldsymbol{y},\lambda^1,\mu^1)\big|_{\boldsymbol{x}^1}^{lin} \\ & g_i^1(\boldsymbol{y}) \leq 0 \quad if\ x_i^{B_j} = x_i^U \\ & g_i^1(\boldsymbol{y}) \geq 0 \quad if\ x_i^{B_j} = x_i^L \end{array}\right\} \forall j \in UL(1,2) \\ \mu_B \geq \min\limits_{\boldsymbol{x}\in X} L(\boldsymbol{x},\boldsymbol{y},\lambda^2,\mu^2) \end{array}\right\} \tag{3.18}$$

Then,

$$\overline{\mu}_B{}^2 \geq \min_{B_l\in CB}\left\{\begin{array}{l} \min\limits_{\boldsymbol{y},\mu_B} \mu_B \\ s.t. \\ \left.\begin{array}{ll} \mu_B \geq & L(\boldsymbol{x}^{B_j},\boldsymbol{y},\lambda^1,\mu^1)\big|_{\boldsymbol{x}^1}^{lin} \\ & g_i^1(\boldsymbol{y}) \leq 0 \quad if\ x_i^{B_j} = x_i^U \\ & g_i^1(\boldsymbol{y}) \geq 0 \quad if\ x_i^{B_j} = x_i^L \end{array}\right\} \forall j \in UL(1,2) \\ \begin{array}{ll} \mu_B \geq & L(\boldsymbol{x}^{B_l},\boldsymbol{y},\lambda^2,\mu^2)\big|_{\boldsymbol{x}^2}^{lin} \\ & g_i^2(\boldsymbol{y}) \leq 0 \quad if\ x_i^{B_l} = x_i^U \\ & g_i^2(\boldsymbol{y}) \geq 0 \quad if\ x_i^{B_l} = x_i^L \end{array} \end{array}\right\} \tag{3.19}$$

Since the first set of constraints for this problem are functions of only $\boldsymbol{y}$, the $\min_{B_l \in CB}$ operator applies only to the second set of constraints, that is, those corresponding to the 2nd primal problem. Hence, (3.19) is equivalent to

$$\overline{\mu}_B{}^2 \geq \left\{ \begin{array}{l} \min\limits_{\boldsymbol{y},\mu_B} \ \mu_B \\ s.t. \\ \mu_B \geq \left. \begin{array}{l} L(\boldsymbol{x}^{B_j}, \boldsymbol{y}, \lambda^1, \mu^1)\big|^{lin}_{\boldsymbol{x}^1} \\ g_i^1(\boldsymbol{y}) \leq 0 \quad if\ x_i^{B_j} = x_i^U \\ g_i^1(\boldsymbol{y}) \geq 0 \quad if\ x_i^{B_j} = x_i^L \end{array} \right\} \forall j \in UL(1,2) \\ \mu_B \geq \min\limits_{B_l \in CB} \left\{ \begin{array}{l} L(\boldsymbol{x}^{B_l}, \boldsymbol{y}, \lambda^2, \mu^2)\big|^{lin}_{\boldsymbol{x}^2} \\ g_i^2(\boldsymbol{y}) \leq 0 \quad if\ x_i^{B_l} = x_i^U \\ g_i^2(\boldsymbol{y}) \geq 0 \quad if\ x_i^{B_l} = x_i^L \end{array} \right. \end{array} \right\} \tag{3.20}$$

The use of Property 7 for the second iteration gives the following result:

$$\min_{\boldsymbol{x} \in X} L(\boldsymbol{x}, \boldsymbol{y}, \lambda^2, \mu^2) \ \geq \ \min_{B_l \in CB} \left\{ \begin{array}{l} L(\boldsymbol{x}^{B_l}, \boldsymbol{y}, \lambda^2, \mu^2) \\ g_i^2(\boldsymbol{y}) \leq 0 \quad if\ x_i^{B_l} = x_i^U \\ g_i^2(\boldsymbol{y}) \geq 0 \quad if\ x_i^{B_l} = x_i^L \end{array} \right\} \tag{3.21}$$

Hence, from (3.20) and (3.21), the second set of constraints on the RHS of (3.19) is simply a relaxed form of the second set of constraints of the RHS of (3.18). Hence, for $k = 2$, the property holds. Similarly, by induction, the property can be proven for any k.

Property 3.3.7 is the final link in the series of rigorous simplifications made in the inner relaxed dual problem. It implies that the relaxed dual problem at the Kth iteration can be replaced by the RHS of (3.17) and solved as a series of convex subproblems.

3.4 INFEASIBLE PRIMAL PROBLEMS

For some values of $\boldsymbol{y} = \boldsymbol{y}^k$, the primal subproblem (3.2) may not have a feasible solution. Note that for unconstrained optimization problems or for problems where it is possible to incorporate the constraint set in the relaxed dual problem (for example, problems with only linear constraints such as those that arise in phase and chemical reaction equilibrium (see also Chapter 6)) we will not have infeasible primal problems. This is also true for problems involving only one joint constraint in $\boldsymbol{x}$ and $\boldsymbol{y}$ such as the illustrative example, where the constraint can be used to generate a bound for the $\boldsymbol{y}$ variable, thus ensuring that it is always feasible *w.r.t.* the primal problem. In the case of infeasible primal problems, a different subproblem must be solved for generating the

appropriate values of λ^k and μ^k. As it is shown in Floudas (1995), one possible formulation for this problem is

$$\begin{aligned} \min_{\substack{\alpha \geq 0 \\ x \in X}} \ \delta &= \sum_{i=1}^{m} \alpha_i \\ \boldsymbol{h}(\boldsymbol{x}, \boldsymbol{y}^k) &= \boldsymbol{0} \\ \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{y}^k) - \alpha &\leq \boldsymbol{0}. \end{aligned} \tag{3.22}$$

In problem (3.22), for every fixed $\boldsymbol{y}^k$, the objective function is linear, the equality constraints are linear and the inequality constraints are convex. Then, if $\overline{\delta}$ is the optimal solution of (3.22), the Strong Duality Theorem (Bazaraa *et al.*, 1993; Floudas, 1995), provides

$$\overline{\delta} = \max_{\substack{\mu_1^k \geq 0 \\ \lambda_1^k}} \min_{x \in X} \left\{ \delta + \lambda_1^{k^T} \boldsymbol{h}(\boldsymbol{x}, \boldsymbol{y}^k) + \mu_1^{k^T} (\boldsymbol{g}(\boldsymbol{x}, \boldsymbol{y}^k) - \alpha) \right\} \tag{3.23}$$

where λ_1^k and μ_1^k are the vectors of Lagrange multipliers. Applying the KKT optimality conditions for (3.22), (3.23) implies that

$$\overline{\delta} = \max_{\substack{\lambda_1^k \\ \mu_1^k \geq 0}} \min_{x \in X} \{ \lambda_1^{k^T} \boldsymbol{h}(\boldsymbol{x}, \boldsymbol{y}) + \mu_1^{k^T} \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{y}) \}.$$

Therefore, since we seek to minimize the infeasibility $\overline{\delta}$, a useful constraint to be added to the relaxed dual problem is

$$\max_{\substack{\lambda_1^k \\ \mu_1^k \geq 0}} \min_{x \in X} \{ \lambda_1^{k^T} \boldsymbol{h}(\boldsymbol{x}, \boldsymbol{y}) + \mu_1^{k^T} \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{y}) \} = 0.$$

A relaxed form of this constraint is

$$\min_{x \in X} L_1(\boldsymbol{x}, \boldsymbol{y}, \lambda_1^k, \mu_1^k) \leq 0 \tag{3.24}$$

where

$$L_1(\boldsymbol{x}, \boldsymbol{y}, \lambda_1^k, \mu_1^k) = \lambda_1^{k^T} \boldsymbol{h}(\boldsymbol{x}, \boldsymbol{y}) + \mu_1^{k^T} \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{y}). \tag{3.25}$$

This is the appropriate constraint to be added to the relaxed dual problem for those iterations where the primal problem is infeasible. It serves as a *feasibility cut*, eliminating the possibility of any relaxed dual problem returning the same value of $\boldsymbol{y}^k$ again. It can be easily shown that the Properties 3.3.2-3.3.7 are all applicable directly for the case of infeasible primal problems by simply replacing μ_B by 0 and $L(\boldsymbol{x}, \boldsymbol{y}, \lambda^k, \mu^k)$ by $L_1(\boldsymbol{x}, \boldsymbol{y}, \lambda_1^k, \mu_1^k)$.

It should be noted that the infeasibility cuts can also be introduced in other forms into the relaxed dual problem. For example, at any iteration, $\mu_B \geq R^{LBD}$, where R^{LBD} is the lower bound obtained from all the *relaxed dual* problems up to that iteration. Using this and (3.24), another constraint that can be added to the relaxed dual problem is

$$\mu_B \geq R^{LBD} + \min_{x} L'(\boldsymbol{x}, \boldsymbol{y}, \lambda_1, \mu_1).$$

3.5 THE GLOBAL OPTIMIZATION (GOP) ALGORITHM

Based on the properties presented in the previous section, the GOP algorithm is presented for solving (3.1). The proof of convergence is presented in Section 3.6. The algorithm can be formally stated in the following steps:

STEP 0 – *(Initialization of parameters)*

Define the parameters $\mu_B^{stor}(K^{max}, B_j)$, $\boldsymbol{y}^{stor}(K^{max}, B_j)$, and $\boldsymbol{y}^k(K^{max}, B_j)$ over the set of bounds CB and the maximum expected number of iterations K^{max}. Define P^{UBD} and R^{LBD} as the upper and lower bounds obtained from the primal and relaxed dual problems respectively. Set

$$\mu_B^{stor}(K^{max}, B_j) = U, \; P^{UBD} = U, \quad and \; R^{LBD} = L$$

where U is a very large *positive* number and L is a very large *negative* number. Select an initial set of values y^1. Set the counter K equal to 1, and sets K^{feas} and K^{infeas} to empty sets. Select a convergence tolerance parameter ϵ.

STEP 1 – *(Primal problem)*

Store the value of $\boldsymbol{y}^K$. Solve the primal problem (3.2) for $\boldsymbol{y} = \boldsymbol{y}^K$. If this problem is feasible, update the set K^{feas} to contain K, and store the optimal Lagrange multipliers λ^K and μ^K. Update the upper bound so that $P^{UBD} = MIN(P^{UBD}, P^K(y^K))$, where $P^K(y^K)$ is the solution of the current (K th) primal problem.

If the primal problem is infeasible, update the set K^{infeas} to contain K, and solve the relaxed primal problem (3.22) for $\boldsymbol{y} = \boldsymbol{y}^K$. Store the values of the optimal Lagrange multipliers λ_1^K and μ_1^K.

STEP 2 – *(Selection of Lagrange functions from the previous iterations)*

For $k = 1, 2, \ldots, K-1$, evaluate all the *qualifying* constraints of every Lagrange function (i.e., corresponding to each set of bounds of $\boldsymbol{x}$) at $\boldsymbol{y}^K$. If every *qualifying* constraint of a Lagrange function is satisfied at $\boldsymbol{y}^K$, select that Lagrange function to be in the set $UL(k, K)$, that is, to be present in the current relaxed dual problems along with its *qualifying* constraints.

STEP 3 – *(Relaxed Dual Problem)*

Formulate the Lagrange function corresponding to the current primal problem. Add this as a constraint to the relaxed dual problem. Then:

(a) Determine the set of *connected* x variables, I_c^K.

(b) Select a combination of the bounds of the *connected* variables in $\boldsymbol{x}$, say $B_l = B_1 \in CB$.

(c) Solve the following *relaxed dual* subproblem:

$$\min_{\boldsymbol{y}\in Y,\mu_B} \mu_B$$

s.t.

$$\left.\begin{array}{ll} \mu_B \geq & L(\boldsymbol{x}^{B_j},\boldsymbol{y},\lambda^k,\mu^k)\big|^{lin}_{\boldsymbol{x}^k} \\ & g_i^k(\boldsymbol{y}) \leq 0 \quad if\ x_i^{B_j} = x_i^U \\ & g_i^k(\boldsymbol{y}) \geq 0 \quad if\ x_i^{B_j} = x_i^L \end{array}\right\} \quad \begin{array}{c} \forall j \in UL(k,K) \\ k = 1,2,\ldots,K-1 \\ k \in K^{feas} \end{array}$$

$$\left.\begin{array}{ll} 0 \geq & L_1(\boldsymbol{x}^{B_j},\boldsymbol{y},\lambda_1^k,\mu_1^k)\big|^{lin}_{\boldsymbol{x}^k} \\ & g_i^k(\boldsymbol{y}) \leq 0 \quad if\ x_i^{B_j} = x_i^U \\ & g_i^k(\boldsymbol{y}) \geq 0 \quad if\ x_i^{B_j} = x_i^L \end{array}\right\} \quad \begin{array}{c} \forall j \in UL(k,K) \\ k = 1,2,\ldots,K-1 \\ k \in K^{infeas} \end{array}$$

$$\left.\begin{array}{ll} \mu_B \geq & L(\boldsymbol{x}^{B_l},\boldsymbol{y},\lambda^K,\mu^K)\big|^{lin}_{\boldsymbol{x}^k} \\ & g_i^K(\boldsymbol{y}) \leq 0 \quad if\ x_i^{B_l} = x_i^U \\ & g_i^K(\boldsymbol{y}) \geq 0 \quad if\ x_i^{B_l} = x_i^L \end{array}\right\} \quad if\ K \in K^{feas}$$

$$\left.\begin{array}{ll} 0 \geq & L_1(\boldsymbol{x}^{B_l},\boldsymbol{y},\lambda_1^K,\mu_1^K)\big|^{lin}_{\boldsymbol{x}^k} \\ & g_i^K(\boldsymbol{y}) \leq 0 \quad if\ x_i^{B_l} = x_i^U \\ & g_i^K(\boldsymbol{y}) \geq 0 \quad if\ x_i^{B_l} = x_i^L \end{array}\right\} \quad if\ K \in K^{infeas}$$

Store the solution, if feasible, in $\mu_B^{stor}(K,B_l)$ and $\boldsymbol{y}^{stor}(K,B_l)$.

(d) Select a new combination of bounds for $\boldsymbol{x}$, say $B_l = B_2$.

(e) Repeat (b)-(d) until the relaxed dual problem has been solved at each set of bounds of the *connected* variables in $\boldsymbol{x}$, that is, for every $B_l \in CB$.

STEP 4 – *(Selecting a new lower bound and $\boldsymbol{y}^{K+1}$)*
From the stored set μ_B^{stor}, select the minimum μ_B^{min} (including the solutions from the current iteration). Also, select the corresponding stored value of $\boldsymbol{y}^{stor}(k,B_j)$ as $\boldsymbol{y}^{min}$. Set $R^{LBD} = \mu_B^{min}$, and $\boldsymbol{y}^{K+1} = \boldsymbol{y}^{min}$. Delete μ_B^{min} and $\boldsymbol{y}^{min}$ from the stored set.

STEP 5 – *(Check for convergence)*
Check if $R^{LBD} > P^{UBD} - \epsilon$. IF yes, STOP. Else, set $K = K+1$ and return to step 1.

A more detailed discussion of the key steps of the algorithm are presented below.

3.5.1 Selection Of Previous Lagrange Functions

At any given iteration, the relaxed dual subproblems will contain a Lagrange function from the current iteration, and one from each of the previous iterations.

The criterion of selection of the Lagrange functions from the previous iterations is very important, since it defines the region in which a particular relaxed dual subproblem is solved. The *qualifying* constraints of the Lagrange functions from previous iterations are used to select the correct Lagrange function from each of these iterations. If the *qualifying* constraints of a Lagrange function are satisfied at y^K, then that function and its accompanying *qualifying* constraints are selected to be constraints for the current relaxed dual subproblems. Exactly one Lagrange function will be selected from each of the previous iterations.

There can be cases, however, when this criterion fails to select exactly one Lagrange function. It is possible (especially in the case of iterations for which the primal problems are infeasible) that at some iteration, say the Kth one, the *qualifying* constraint of a Lagrange function from a previous iteration (say for the kth iteration) with respect to a particular x_j may be strictly satisfied as an equality. That is,

$$g_j^k(\boldsymbol{y}^K) = 0$$

implying that all the Lagrange functions from the kth iteration with x_j set to either its lower or upper bound can be potentially present in the current relaxed dual subproblems. Thus, there will be two Lagrange functions from the kth iteration present in the relaxed dual subproblems of the current (Kth) iteration, along with their *qualifying* constraints, which are

$$\begin{aligned} g_j^k(\boldsymbol{y}) &\geq 0 \quad if \;\; x_j^B = x_j^L \\ g_j^k(\boldsymbol{y}) &\leq 0 \quad if \;\; x_j^B = x_j^U. \end{aligned}$$

Together, these constraints imply that

$$g_j^k(\boldsymbol{y}) = 0.$$

This reduces the degrees of freedom for the current relaxed dual subproblems by one, and could potentially lead to some regions of $\boldsymbol{y}$ being unavailable for the relaxed dual subproblems. However, this can be avoided if the *qualifying* constraints are introduced in a perturbed form as shown below :

$$\begin{aligned} g_j^k(\boldsymbol{y}) &\geq \delta \quad if \;\; x_j^B = x_j^L \\ g_j^k(\boldsymbol{y}) &\leq -\delta \quad if \;\; x_j^B = x_j^U \end{aligned}$$

where δ is a very small positive number. This ensures that when these *qualifying* constraints are checked at a later iteration, they cannot both be simultaneously satisfied. Hence, only one of the two corresponding Lagrange functions can be selected. The parameter δ can be made sufficiently small so that it does not significantly affect the optimal solution found by the relaxed dual subproblems.

3.5.2 Partition in the $\boldsymbol{y}$-space

Once the Lagrange functions from the previous iterations have been selected, the relaxed dual subproblems are then solved, one for each combination of

x^{B_l}. In each case, the Lagrange function formulated from the *current* primal problem (along with its *qualifying* constraints) is chosen as a constraint for the relaxed dual problem. Since the relaxed dual subproblems are solved for different combinations of bounds, each *relaxed dual* subproblem has a unique set of *qualifying* constraints associated with it. Therefore, each of these *relaxed dual* subproblems solved corresponds to a different region in the y-space of the original problem. Consequently, the solution of the *relaxed dual* subproblems can be viewed as a branch-and-bound procedure where the feasible region in terms of the y variables is partitioned into $2^{NI_c^K}$ subregions (where NI_c^K is the number of *connected* x variables at the Kth iteration) and each *relaxed dual* subproblem corresponds to a node of the branch-and-bound tree.

3.5.3 Selection of the Lower Bound

After the relaxed dual problem has been solved for every possible combination of the bounds x_l^B, the only remaining task is to determine a new lower bound for the original problem and select a fixed value of y for the next primal problem. This is done by taking the minimum of all the stored values of μ_B as the lower bound, and the corresponding solutions for y as the y^{K+1} for the next primal. Once a particular μ_B and y have been selected, they are no longer considered for future iterations. This is to ensure that the relaxed dual problem will not return the same value of y and μ_B during successive iterations.

Finally, the check for convergence is done. If the lower bound from the relaxed dual problem comes within ϵ of the upper bound from the primal problems, the optimal solution to the original problem has been found and the algorithm terminates. Otherwise, the algorithm continues with the updated stored set of solutions from the relaxed dual subproblems and a new fixed value of y for the primal problem.

3.6 FINITE ϵ-CONVERGENCE AND ϵ-GLOBAL OPTIMALITY

This section presents the formal theoretical proof of the convergence and global optimality of the GOP algorithm. The convergence proof is based upon the results of Geoffrion (1972).

Lemma 3.6.1 *If the sequence of solutions $< y^k >$ of the relaxed dual problem converges to $\overline{y}$, and the corresponding Lagrange multipliers for the primal problem (3.2) converge to $(\overline{\lambda}, \overline{\mu})$, then*

$$L(x^{B_j}, \overline{y}, \overline{\lambda}, \overline{\mu})\big|_{x^k}^{lin} = P(\overline{y}) \quad for\ any\ x^{B_j}$$

where $P(\overline{y})$ represents the optimal solution of the primal problem at $\overline{y}$.

Proof. The linearization of the Lagrange function evaluated at x^{B_j} can be written as

$$L(x^{B_j}, y, \lambda^k, \mu^k)\big|_{x^k}^{lin} = L(x^k, y, \lambda^k, \mu^k) + [g^k(y) - g^k(y^k)] \cdot (x^{B_j} - x^k) \quad \forall B_j \in CB$$

By the Strong Duality Theorem, $L(\boldsymbol{x}^k, \boldsymbol{y}^k, \lambda^k, \mu^k)$ is equal to $P(\boldsymbol{y}^k)$ for every $\boldsymbol{y}^k$. Then,

$$As \quad \boldsymbol{y}^k \to \overline{\boldsymbol{y}}, \quad L(\boldsymbol{x}^k, \boldsymbol{y}, \lambda^k, \mu^k)\big|_{\boldsymbol{y}=\overline{\boldsymbol{y}}} \to P(\overline{\boldsymbol{y}}). \tag{3.26}$$

Since $\boldsymbol{g}^k(\boldsymbol{y})$ is a vector of linear functions of $\boldsymbol{y}$,

$$As \quad \boldsymbol{y}^k \to \overline{\boldsymbol{y}}, \quad \boldsymbol{g}^k(\boldsymbol{y}) \to \boldsymbol{g}(\overline{\boldsymbol{y}}).$$

From this, and (3.26), we obtain the desired result.

Lemma 3.6.2 *For a fixed $\boldsymbol{y} = \boldsymbol{y}^k$ for the kth primal problem, let $U(\boldsymbol{y}^k)$ be the set of optimal multipliers. If $\boldsymbol{y}^k \to \overline{\boldsymbol{y}}$ and $\lambda^k \to \overline{\lambda}, \mu^k \to \overline{\mu}$, then $(\overline{\lambda}, \overline{\mu}) \in U(\overline{\boldsymbol{y}})$.*

Proof. The proof for this Lemma comes from showing that $U(\boldsymbol{y})$ is an upper semi-continuous mapping at $\overline{\boldsymbol{y}}$. To do this, we employ the characterization of $U(\boldsymbol{y})$ as the set of optimal solutions of the dual of the primal problem; that is ,

$$U(\boldsymbol{y}) = \{\mu \geq 0, \lambda : \; L^*(\overline{\boldsymbol{x}}, \boldsymbol{y}, \lambda, \mu) = \max_{\mu^k \geq 0, \lambda^k} L^*(\overline{\boldsymbol{x}}, \boldsymbol{y}, \lambda^k, \mu^k)\}$$

Now, $L^*(\overline{\boldsymbol{x}}, \boldsymbol{y}, \lambda^k, \mu^k) = \min_{\boldsymbol{x}} L(\boldsymbol{x}, \boldsymbol{y}, \lambda^k, \mu^k)$ is a continuous function, since it is a linear sum of continuous functions of $\boldsymbol{y}$. Then, application of Theorem 1.5 of (Meyer, 1970) proves the desired result.

Lemma 3.6.3 *Let $P(\boldsymbol{y})$ be the set of optimal solutions of the primal problem for fixed values of $\boldsymbol{y}$; then, if $\boldsymbol{y}^k \to \overline{\boldsymbol{y}}$, $P(\boldsymbol{y})$ is upper semi-continuous at $\overline{\boldsymbol{y}}$.*

Proof. Since the set $U(\boldsymbol{y})$ is assumed to be nonempty for all $\boldsymbol{y}$, then from the Strong Duality Theorem, we have that

$$P(\boldsymbol{y}) = \max_{\mu \geq 0, \lambda} L^*(\overline{\boldsymbol{x}}, \boldsymbol{y}, \lambda, \mu) \quad \forall \boldsymbol{y} \in Y$$

Since Y is a compact set, the local uniform boundedness of $U(\boldsymbol{y})$ implies the uniform boundedness of $U(\boldsymbol{y})$ on all of Y. Therefore, there exists a compact set U^* such that $U(\boldsymbol{y}) \subseteq U^*$ for all $\boldsymbol{y}$ in Y. The constraint $(\lambda, \mu) \in U^*$ can then be added without disturbing the equality. Since $L^*(\overline{\boldsymbol{x}}, \boldsymbol{y}, \lambda, \mu)$ is a continuous function, $P(\boldsymbol{y})$ is upper semi-continuous at $\overline{\boldsymbol{y}}$ (Maximum Theorem of Berge, 1963).

Theorem 3.6.1 (Finite ϵ-convergence) *If the following conditions hold:*

(a) *X and Y are nonempty compact convex sets, and $Y \subset V$,*

(b) *Conditions (A),*

(c) *$f(\boldsymbol{x}, \boldsymbol{y})$, $\boldsymbol{g}(\boldsymbol{x}, \boldsymbol{y})$ and $\boldsymbol{h}(\boldsymbol{x}, \boldsymbol{y})$ are continuous on $X \times Y$, and*

(d) *The set $U(y)$ of optimal multipliers for the primal problem is nonempty for all $y \in Y$ and uniformly bounded in some neighborhood of every such point,*

then,
For any given $\epsilon > 0$, the GOP algorithm terminates in a finite number of steps.

Proof. Fix $\epsilon > 0$ arbitrarily. Suppose that the GOP algorithm does not converge in a finite number of iterations (that is, $(\mu_B^{min^k}) < P^{UBD} - \epsilon$). Let $< y^k, (\mu_B^{min^k}) >$ be the sequence of optimal solutions to the relaxed dual problem at successive iterations k. Note that any solution from the stored set, if selected as the minimum for a given iteration, is removed from the stored set. Therefore, by taking a subsequence, if necessary, we may assume that $< y^k, (\mu_B^{min^k}) >$ converges to $(\overline{y}, \overline{\mu_B^{min}})$ such that $\overline{y} \in Y$. At every iteration, there is an accumulation of constraints from previous iterations. This implies that μ_B^{min} is a nondecreasing sequence which is bounded above by the optimal value of the original problem. Also, at every iteration, y^k is in the compact set Y. Similarly, since $U(y)$ is uniformly bounded for all $y \in Y$, we may assume that the corresponding sequence of multipliers for the primal problems (λ^k, μ^k) converges to $(\overline{\lambda}, \overline{\mu})$, and that the solutions of the corresponding primal problems $(x^k, P(y^k))$ converge to $(\overline{x}, P(\overline{y}))$. From Lemma 3.6.2, we have that $(\overline{\lambda}, \overline{\mu}) \in U(\overline{y})$. From Lemma 3.6.1,

$$L(x^{B_j}, \overline{y}, \overline{\lambda}, \overline{\mu})\big|_{x^k}^{lin} = P(\overline{y}) \quad for\ any\ x^{B_j}. \tag{3.27}$$

Now, at every iteration k, due to accumulation of constraints,

$$\mu_B^{min^k} \geq L(x^{B_j}, y^{k+1}, \lambda^k, \mu^k)\big|_{x^k}^{lin}$$

for some combination of bounds x^{B_j}. Hence, by continuity of $L(x^{B_j}, y, \lambda^k, \mu^k)|_{x^k}^{lin}$ and (3.27), we obtain $\overline{\mu_B^{min}} \geq P(\overline{y})$. Then, the upper semi-continuity of $P(y)$ at $\overline{y}$ (Lemma 3.6.3) implies that $\mu_B^{min^k} \geq P(y^k) - \epsilon$ for all k sufficiently large, which contradicts the assumption that the termination criterion in Step 5 of the GOP algorithm (Section 3.5) is never met.

Remark 3.6.1 It should be noted that in the case that Y is not a subset of V then only asymptotic ϵ-convergence is achieved.

Theorem 3.6.2 (Global Optimality)
If the conditions stated in Theorem 3.6.1 hold, then

(i) *the solution of the* relaxed dual *problem in Step 3 of the GOP algorithm (Section 3.5) will always be a valid underestimator of the solution of problem (3.1).*

(ii) *the GOP algorithm will terminate at the global optimum of (3.1).*

Proof.

(i) From Property 3.3.7, the solution of the relaxed dual problem in Step 3 of the GOP algorithm (Section 3.5) will underestimate the solution of (3.5). Since (3.5) has fewer constraints than (3.4), it represents a lower bound on the solution of (3.1). Hence, the solution of the relaxed dual problem in Step 3 will always be a valid underestimator of the optimal solution of (3.1).

(ii) The primal problem at every iteration represents an upper bound for the original problem (3.1), while the relaxed dual problem contains fewer constraints than the original problem and thus represents a valid lower bound on the solution of (3.1). Therefore, since the termination of the algorithm is based on the difference between the lowest upper bound (from the primal problems) and the largest lower bound (from the relaxed dual problems), the algorithm will terminate when these two bounds are both within ϵ of the solution of (3.1). From Theorem 3.6.1, the algorithm terminates in a finite number of steps. Hence, the GOP algorithm terminates at an ϵ-global optimum of (3.1).

3.7 ILLUSTRATIVE EXAMPLE

We study the application of the GOP algorithm to the example that has been used throughout the theoretical development. Figure 3.2 shows the projection of the problem in the space of y. The function $v(y)$ is obtained by solving the primal problem all possible values of y. As can be seen, the problem has its global solution at $x = 0.5, y = 8$ with an objective value of -8.5. There is also a local minimum at $x = 4, y = 1$ with an objective value of -5, and a local maximum at $x = y = 2$. It is thus clear why this problem, despite having a very simple structure, is hard to solve using a local optimization routine like MINOS (Murtagh and Saunders, 1983).

Consider the starting point of $y = 2$ for the application of the GOP algorithm to the example.

Iteration 1

With $y^1 = 2$, the primal problem is given by

$$\begin{aligned} \min_{x,y} \; & -x - 2 \\ s.t. \quad & 2x \leq 4 \\ & 0 - x \leq 0 \\ & x - 4 \leq 0. \end{aligned}$$

The solution of this problem is $x = 2$, $\mu_1^1 = 0.5$, $\mu_2^1 = 0$, and $\mu_3^1 = 0$. The objective function value of -4 provides the first upper bound on the global solution.

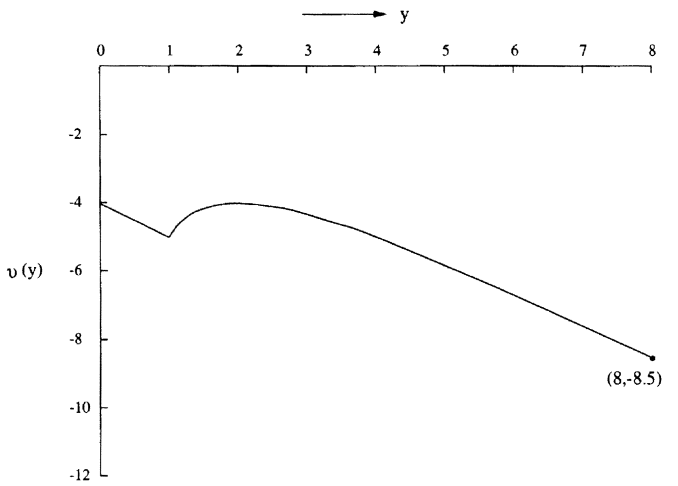


Figure 3.2. Optimal Solution for the Illustrative Example

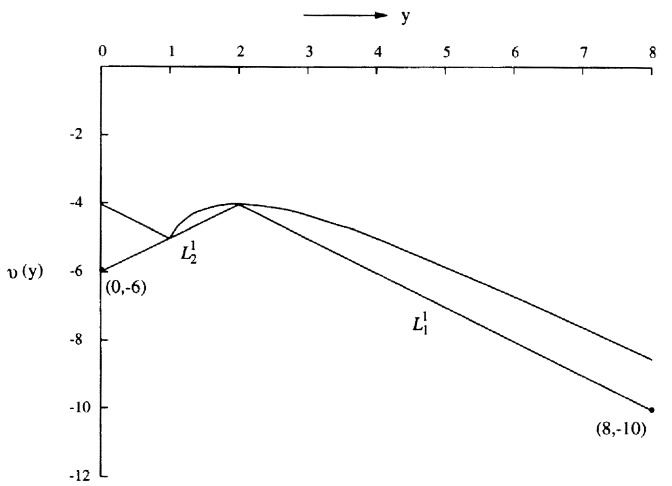


Figure 3.3. Iteration I for the Illustrative Example

The Lagrange function from this problem is

$$\begin{aligned} L(x, y, \mu^1) &= -x - y + \mu_1^1(xy - 4) + \mu_2^1(0 - x) + \mu_3^1(x - 4) \\ &= 0.5(y - 2)x - y - 2 \end{aligned}$$

There is one *connected* variable, and therefore 2 *relaxed dual* subproblems are solved. These are shown below:

(i) For $y - 2 \geq 0$, and $x^B = x^L = 0$:

$$\begin{aligned} \min_{y,\mu_B} \quad & \mu_B \\ s.t. \quad \mu_B &\geq L_1^1 = -y - 2 \\ y - 2 &\geq 0 \\ 0 \leq y &\leq 8. \end{aligned}$$

The solution of this problem is $y = 8$, $\mu_B = -10$.

(ii) For $y - 2 \leq 0$, and $x^B = x^U = 4$:

$$\begin{aligned} \min_{y,\mu_B} \quad & \mu_B \\ s.t. \quad \mu_B &\geq L_2^1 = y - 6 \\ y - 2 &\geq 0 \\ 0 \leq y &\leq 8. \end{aligned}$$

The solution of this problem is $y = 0$, $\mu_B = -6$.

Figure 3.3 shows the Lagrange functions L_1^1 and L_2^1 associated with these two problems. They underestimate the optimal solution for $y \geq 2$ and $y \leq 2$ respectively. Moreover, it can be seen that they intersect at $y = 2$ because the strong duality theorem is satisfied.

Thus, after the first iteration, there are two solutions of (μ_B, y) to choose from. The solution corresponding to the minimum μ_B is chosen. In this case, this corresponds to the solution $\mu_B = -10$, $y = 8$. Hence, at the end of the first iteration, the lower bound is -10, with $y^2 = 8$ for the second iteration.

It is interesting to note the consequence of using the *GOS* (or v3-GBD, Floudas (1995)) technique (Floudas *et al.*, 1989) to solve this problem. The first primal problem is solved as in the GOP algorithm. However, in the Lagrange function formulated from this problem, the x variable is fixed at its solution in the primal problem, that is $x = 2$. This leads to

$$L(x = 2, y, \mu^1) = 0.5(y - 2)2 - y - 2 = -4.$$

From, this, it can be seen that the value of μ_B for the (GOS) master (dual) problem has to be greater than -4. As a result, the (GOS) technique will converge after only one iteration at the suboptimal solution of $x = 2$, $y = 2$.

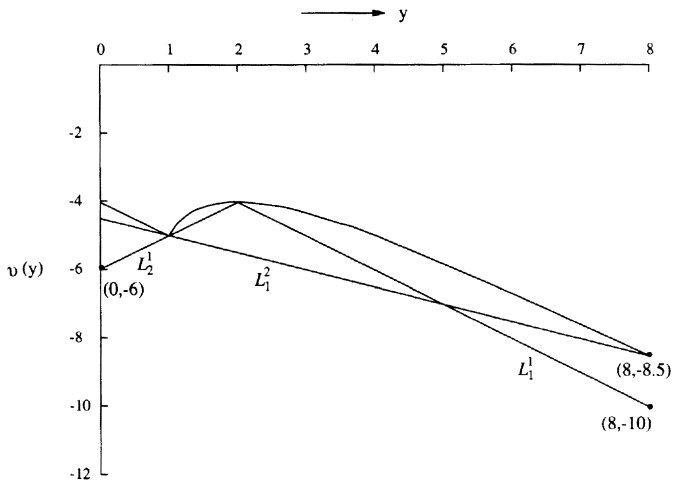


Figure 3.4. Iteration II for the Illustrative Example

Iteration 2

For the second iteration (with $y = 8$), the primal problem is :

$$\begin{aligned} \min_{x,y} \; & -x - 2 \\ s.t. \quad & 8x \leq 4 \\ & 0 - x \leq 0 \\ & x - 4 \leq 0. \end{aligned}$$

This problem has a solution of $x = 0.5$, $\mu_1^2 = 0.125$, $\mu_2^2 = 0$, and $\mu_3^2 = 0$. The objective function value is -8.5, and thus provides a tighter upper bound than the solution from the first iteration.

The Lagrange function formulated from the second primal problem is

$$L(x, y, \mu^2) = -x - y + 0.125(xy - 4) = 0.125(y - 8)x - y - 0.5.$$

Thus, two relaxed dual problems need to be solved: one for $y \leq 8$ and one for $y \geq 8$. However, since $y^U = 8$, we only need to solve for $y \leq 8$.

The Lagrange function from the first iteration will also be present in the current *relaxed dual* problems. The bound used for x in this function (i.e., the choice of either L_1^1 or L_2^1) is made based upon the *qualifying* constraints of L_1^1 and L_2^1. In other words, we check to see if the current point ($y = 8$) satisfies $y \geq 2$ or $y \leq 2$. Since $y^2 \geq 2$, the corresponding Lagrange function from the first iteration, L_1^1, is selected as a constraint. As seen graphically in Figure 3.3, L_1^1 underestimates the optimal solution for all values of $2 \leq y \leq 8$, while L_2^1 underestimates the optimal value of the original problem only for $y \leq 2$ and hence cannot be present for this relaxed dual problem.

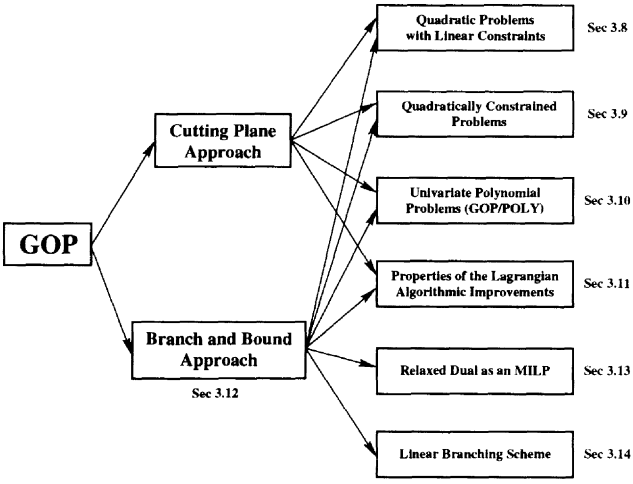


Figure 3.5. Structure of theoretical improvements of GOP

The relaxed dual problem for the second iteration is solved by setting $x = 4$ (corresponding to $y \leq 8$):

$$\min_{y,\mu_B} \ \mu_B$$

$$\begin{aligned} \textit{subject to} \quad \mu_B &\geq L_1^1 = -y - 2 \\ y - 2 &\geq 0 \\ \mu_B &\geq L_2^2 = -0.5y - 4.5 \\ y - 8 &\leq 0 \\ 0 \leq y &\leq 8. \end{aligned}$$

The solution to this problem is $\mu_B = -8.5$, $y = 8$. Figure 3.4 shows the relaxed dual problem for the second iteration pictorially. Both L_1^1 and L_2^2 achieve a minimum at $y = 8$.

At the end of this iteration, there are two solutions to choose from - $\mu_B = -6, y = 0$ and $\mu_B = -8.5, y = 8$. Thus, the lower bound obtained is -8.5 and the algorithm terminates.

In the following sections, we focus on the theory of GOP for quadratic problems with linear constraints, quadratically constrained problems, univariate polynomial problems, and we discuss additional properties of the Lagrange function that lead to algorithmic improvements. Furthermore, we present the GOP from the cutting plane perspective, as well as the branch and bound point of view. Figure 3.5 depicts these aforementioned improvements of the GOP.

3.8 THE GOP FOR QUADRATIC PROBLEMS WITH LINEAR CONSTRAINTS

The general quadratic programming problem, with linear constraints, has the following form:

$$\min_{\boldsymbol{x}\in X} \ \boldsymbol{c}^T\boldsymbol{x} + \boldsymbol{x}^T\boldsymbol{Q}\boldsymbol{x}$$

$$\begin{aligned} s.t. \qquad \boldsymbol{A}_1\boldsymbol{x} - \boldsymbol{b}_1 &\leq \boldsymbol{0} \\ \boldsymbol{A}_2\boldsymbol{x} - \boldsymbol{b}_2 &= \boldsymbol{0} \end{aligned}$$

where $\boldsymbol{x}$ an n-vector of variables, and $\boldsymbol{c}$, $\boldsymbol{b}_1$ and $\boldsymbol{b}_2$ are constant vectors. $\boldsymbol{Q}$, $\boldsymbol{A}_1$ and $\boldsymbol{A}_2$ are constant matrices. Depending on the nature of the eigenvalues of the matrix $\boldsymbol{Q}$, the problem can be a negative definite or indefinite quadratic programming problem.

By defining a new set of variables $\boldsymbol{y} = \boldsymbol{x}$, and introducing a set of equality constraints, this problem can be converted to the following equivalent problem:

$$\min_{\boldsymbol{x},\boldsymbol{y}} \ \boldsymbol{c}^T\boldsymbol{x} + \boldsymbol{x}^T\boldsymbol{Q}\boldsymbol{y}$$

$$\begin{aligned} s.t \quad \boldsymbol{A}_1\boldsymbol{y} - \boldsymbol{b}_1 &\leq \boldsymbol{0} \\ \boldsymbol{A}_2\boldsymbol{y} - \boldsymbol{b}_2 &= \boldsymbol{0} \\ \boldsymbol{x} - \boldsymbol{y} &= \boldsymbol{0}. \end{aligned}$$

By projecting on $\boldsymbol{y}$, the following linear primal problem is obtained at the kth iteration:

$$\min_{\boldsymbol{x}} \ \boldsymbol{c}^T\boldsymbol{x} + \boldsymbol{x}^T\boldsymbol{Q}\boldsymbol{y}^k$$

$$\begin{aligned} s.t \qquad \boldsymbol{A}_1\boldsymbol{x} - \boldsymbol{b}_1 &\leq \boldsymbol{0} \\ \boldsymbol{A}_2\boldsymbol{x} - \boldsymbol{b}_2 &= \boldsymbol{0} \\ \boldsymbol{x} - \boldsymbol{y}^k &= \boldsymbol{0}. \end{aligned}$$

The solution to this problem provides the optimal multiplier vectors μ^k, λ^k corresponding to the original inequality and equality constraints, and the multipliers ν^k for the equality constraints due to the introduction of the $\boldsymbol{y}$-variables.

The Lagrange function can be formulated as

$$\begin{aligned} L(\boldsymbol{x},\boldsymbol{y},\lambda^k,\mu^k,\nu^k) &= \boldsymbol{c}^T\boldsymbol{x} + \boldsymbol{x}^T\boldsymbol{Q}\boldsymbol{y} + {\mu^k}^T(\boldsymbol{A}_1\boldsymbol{x} - \boldsymbol{b}_1) \\ &+ {\lambda^k}^T(\boldsymbol{A}_2\boldsymbol{x} - \boldsymbol{b}_2) + {\nu^k}^T(\boldsymbol{x} - \boldsymbol{y}) \\ &= \boldsymbol{x}^T\boldsymbol{Q}\boldsymbol{y} + \boldsymbol{x}^T[\boldsymbol{c} + \boldsymbol{A}_1^T\mu^k + \boldsymbol{A}_2^T\lambda^k + \nu^k] \\ &- [{\mu^k}^T\boldsymbol{b}_1 + {\lambda^k}^T\boldsymbol{b}_2 + {\nu^k}^T\boldsymbol{y}]. \end{aligned} \tag{3.28}$$

From the KKT gradient condition for the primal problem we have

$$\nabla_{\boldsymbol{x}} L(\boldsymbol{x}^k, \boldsymbol{y}^k, \lambda^K, \mu^k, \nu^k) = \boldsymbol{Q}\boldsymbol{y}^k + \boldsymbol{c} + \boldsymbol{A}_1{}^T\mu^k + \boldsymbol{A}_2^T\lambda^k + \nu^k = \boldsymbol{0}.$$

Hence,

$$\boldsymbol{c} + \boldsymbol{A}_1^T\mu^k + \boldsymbol{A}_2^T\lambda^k + \nu^k \;=\; -\boldsymbol{Q}\boldsymbol{y}^k. \tag{3.29}$$

Thus, (3.28) and (3.29) can be combined to give

$$\begin{aligned} L(\boldsymbol{x}, \boldsymbol{y}, \lambda^k, \mu^k, \nu^k) &= \boldsymbol{x}^T(\boldsymbol{Q}\boldsymbol{y} - \boldsymbol{Q}\boldsymbol{y}^k) - (\mu^{k^T}\boldsymbol{b}_1 + \lambda^{k^T}\boldsymbol{b}_2 + \nu^{k^T}\boldsymbol{y}) \\ &= \boldsymbol{x}^T\boldsymbol{Q}(\boldsymbol{y} - \boldsymbol{y}^k) - (\mu^{k^T}\boldsymbol{b}_1 + \lambda^{k^T}\boldsymbol{b}_2 + \nu^{k^T}\boldsymbol{y}). \end{aligned}$$

Hence, the *qualifying* constraints to be added along with the Lagrange function in the relaxed dual problem take the form

$$\begin{aligned} Q_i(\boldsymbol{y} - \boldsymbol{y}^k) &\leq 0 \quad if \;\; x_i^B = x_i^U \\ Q_i(\boldsymbol{y} - \boldsymbol{y}^k) &\geq 0 \quad if \;\; x_i^B = x_i^L \end{aligned}$$

where Q_i is the ith row of $\boldsymbol{Q}$, that is, the row corresponding to x_i in the matrix $\boldsymbol{Q}$.

For iterations where the primal problem is infeasible, the relaxed primal problem is linear, and hence the Lagrange function generated from this problem (by use of the KKT optimality conditions for the relaxed primal problem) does not depend on $\boldsymbol{x}$. Therefor, for such iterations, only one *relaxed dual* subproblem needs to be solved.

3.9 THE GOP FOR QUADRATICALLY CONSTRAINED PROBLEMS

The quadratic programming problem with quadratic constraints has the following form:

$$\min_{\boldsymbol{x}} \; \boldsymbol{c}^T\boldsymbol{x} + \boldsymbol{x}^T\boldsymbol{Q}\boldsymbol{x}$$

$$\begin{aligned} s.t. \qquad \boldsymbol{x}^T\boldsymbol{A}_m\boldsymbol{x} + \boldsymbol{B}\boldsymbol{x} - \boldsymbol{b}_m &\leq \boldsymbol{0} \qquad m = 1, 2, \ldots, p \\ \boldsymbol{x}^T\boldsymbol{A}_m\boldsymbol{x} + \boldsymbol{C}\boldsymbol{x} - \boldsymbol{b}_m &= \boldsymbol{0} \qquad m = p+1, p+2, \ldots, p+q \\ \boldsymbol{D}\boldsymbol{x} - \boldsymbol{d} &\leq \boldsymbol{0} \\ \boldsymbol{E}\boldsymbol{x} - \boldsymbol{e} &= \boldsymbol{0} \end{aligned} \tag{3.30}$$

where $\boldsymbol{x}$ an n-vector of variables, and $\boldsymbol{c}$, $\boldsymbol{d}$ and $\boldsymbol{e}$ are constant vectors. $\boldsymbol{Q}$, $\boldsymbol{B}$, $\boldsymbol{C}$, $\boldsymbol{D}$, and $\boldsymbol{E}$ are constant matrices. $\boldsymbol{A}_m$ is an $n \times n$ matrix corresponding to the mth quadratic constraint, and $\boldsymbol{b}_m$ is a constant vector for that constraint.

By defining a new set of variables $\boldsymbol{y} = \boldsymbol{x}$, this problem can be converted to the following equivalent problem:

$$\min_{\boldsymbol{x}, \boldsymbol{y}} \; \boldsymbol{c}^T\boldsymbol{x} + \boldsymbol{x}^T\boldsymbol{Q}\boldsymbol{y}$$

$$
\begin{array}{lrcll}
s.t. & \boldsymbol{x}^T\boldsymbol{A}_m\boldsymbol{y} + \boldsymbol{B}\boldsymbol{x} - \boldsymbol{b}_m & \leq & \boldsymbol{0} & m = 1,2,\ldots,p \\
 & \boldsymbol{x}^T\boldsymbol{A}_m y + \boldsymbol{C}\boldsymbol{x} - \boldsymbol{b}_m & = & \boldsymbol{0} & m = p+1, p+2, \ldots, p+q \\
 & \boldsymbol{D}\boldsymbol{x} - \boldsymbol{d} & \leq & \boldsymbol{0} & \\
 & \boldsymbol{E}\boldsymbol{x} - \boldsymbol{e} & = & \boldsymbol{0} & \\
 & \boldsymbol{x} - \boldsymbol{y} & = & \boldsymbol{0}. &
\end{array}
$$

By projecting on $\boldsymbol{y}$, the following linear primal problem is obtained at the kth iteration:

$$
\min_{\boldsymbol{x}} \ \boldsymbol{c}^T\boldsymbol{x} + \boldsymbol{x}^T\boldsymbol{Q}\boldsymbol{y}^k
$$

$$
\begin{array}{lrcll}
s.t. & \boldsymbol{x}^T\boldsymbol{A}_m\boldsymbol{y}^k + \boldsymbol{B}\boldsymbol{x} - \boldsymbol{b}_m & \leq & \boldsymbol{0} & m = 1,2,\ldots,p \\
 & \boldsymbol{x}^T\boldsymbol{A}_m\boldsymbol{y}^k + \boldsymbol{C}\boldsymbol{x} - \boldsymbol{b}_m & = & \boldsymbol{0} & m = p+1, p+2, \ldots, p+q \\
 & \boldsymbol{D}\boldsymbol{x} - \boldsymbol{d} & \leq & \boldsymbol{0} & \\
 & \boldsymbol{E}\boldsymbol{x} - \boldsymbol{e} & = & \boldsymbol{0} & \\
 & \boldsymbol{x} - \boldsymbol{y}^k & = & \boldsymbol{0}. &
\end{array}
$$

The solution to this problem provides the optimal multiplier vectors μ_1^k and λ_1^k corresponding to the quadratic inequality and equality constraints, the multiplier vectors μ_2^k and λ_2^k corresponding to the linear inequality and equality constraints, and the multipliers ν^k for the linear equality constraints due to the introduction of the $\boldsymbol{y}$-variables.

The Lagrange function can be formulated as

$$
\begin{aligned}
L(\boldsymbol{x}, \boldsymbol{y}, \lambda_1^k, \mu_1^k, \lambda_2^k, \mu_2^k, \nu^k) &= \boldsymbol{c}^T\boldsymbol{x} + \boldsymbol{x}^T\boldsymbol{Q}\boldsymbol{y} + \sum_{m=1}^{p} \mu_{1_m}^k (\boldsymbol{x}^T\boldsymbol{A}_m\boldsymbol{y} - \boldsymbol{b}_m) \\
&+ \mu_1^{k^T}\boldsymbol{B}\boldsymbol{x} + \sum_{m=p+1}^{p+q} \lambda_{1_m}^k (\boldsymbol{x}^T\boldsymbol{A}_m\boldsymbol{y} - \boldsymbol{b}_m) \quad (3.31) \\
&+ \lambda_1^{k^T}\boldsymbol{C}\boldsymbol{x} + \mu_2^{k^T}(\boldsymbol{D}\boldsymbol{x} - \boldsymbol{d}) \\
&+ \lambda_2^{k^T}(\boldsymbol{E}\boldsymbol{x} - \boldsymbol{e}) + \nu^{k^T}(\boldsymbol{x} - \boldsymbol{y}).
\end{aligned}
\tag{3.32}
$$

The KKT gradient condition for this problem is

$$
\begin{aligned}
\nabla_{\boldsymbol{x}} L(\boldsymbol{x}^k, \boldsymbol{y}^k, \lambda_1^k, \mu_1^k, \lambda_2^k, \mu_2^k, \nu^k) &= \boldsymbol{Q}\boldsymbol{y}^k + \boldsymbol{c} + \sum_{m=1}^{p} \mu_{1_m}^k \boldsymbol{A}_m\boldsymbol{y}^k + \boldsymbol{B}^T\mu_1^k \\
&+ \sum_{m=p+1}^{p+q} \lambda_{1_m}^k \boldsymbol{A}_m\boldsymbol{y}^k + \boldsymbol{C}^T\lambda_1^k \\
&+ \boldsymbol{D}^T\mu_2^k + \boldsymbol{E}^T\lambda_2^k + \nu^k \\
&= \boldsymbol{0}
\end{aligned}
$$

Hence,

$$\begin{aligned} \boldsymbol{c} + \boldsymbol{B}^T\mu_1^k + \boldsymbol{C}^T\lambda_1^k + \boldsymbol{D}^T\mu_2^k + \boldsymbol{E}^T\lambda_2^k + \nu^k &= -\boldsymbol{Q}\boldsymbol{y}^k - \sum_{m=1}^{p} \mu_{1_m}^k \boldsymbol{A}_m \boldsymbol{y}^k \\ &- \sum_{m=p+1}^{p+q} \lambda_{1_m}^k \boldsymbol{A}_m \boldsymbol{y}^k. \end{aligned} \tag{3.33}$$

Using (3.32) and (3.33), the Lagrange function can be reformulated as :

$$\begin{aligned} L(\boldsymbol{x}, \boldsymbol{y}, \lambda_1^k, \mu_1^k, \lambda_2^k, \mu_2^k, \nu^k) &= \boldsymbol{x}^T(\boldsymbol{Q} + (\mu_1^{k^T}, \lambda_1^{k^T})\boldsymbol{A})(\boldsymbol{y} - \boldsymbol{y}^k)^T \\ &\quad -(\mu_1^{k^T}, \lambda_1^{k^T})\boldsymbol{b} - \mu_2^{k^T}\boldsymbol{d} - \lambda_2^{k^T}\boldsymbol{e} - \nu^{k^T}\boldsymbol{y}. \end{aligned}$$

Hence, the constraints to be added along with the Lagrange function in the relaxed dual problem take the form

$$\begin{aligned} (Q_i + \sum_{m=1}^{p} \mu_{1_m}^k A_{m_i} + \sum_{m=p+1}^{p+q} \lambda_{1_m}^k A_{m_i}).(y - y^k) &\leq 0 \qquad if \quad x_i^B = x_i^U \\ (Q_i + \sum_{m=1}^{p} \mu_{1_m}^k A_{m_i} + \sum_{m=p+1}^{p+q} \lambda_{1_m}^k A_{m_i}).(y - y^k) &\geq 0 \qquad if \quad x_i^B = x_i^L \end{aligned}$$

where the subscript i refers to the ith row of the matrices $\boldsymbol{Q}$ and $\boldsymbol{A}_m$, that is, the rows corresponding to the variable x_i in the matrices. Similarly, the *qualifying* constraints for the infeasible primal problem can be generated.

3.10 THE GOP FOR UNIVARIATE POLYNOMIAL PROBLEMS

The univariate polynomial problems addressed in this section take the form:

$$\min_{y} \quad f(y) = a_0 + a_1 y + a_2 y^2 \; + \; \cdots + a_N y^N$$

$s.t.$

$$\begin{aligned} A_{j0} + A_{j1}y + A_{j2}y^2 + \cdots + A_{jN}y^N &\leq 0 \quad \forall\, j = 1, 2, \ldots, J \\ B_{m0} + B_{m1}y + B_{m2}y^2 + \cdots + B_{mN}y^N &= 0 \quad \forall\, m = 1, 2, \ldots, M \\ y^L \leq y &\leq y^U \end{aligned} \tag{3.34}$$

where y is a *single* variable and A_{ji}, A_{mi} are the coefficients of y^i in the jth inequality and mth equality constraint respectively. The nonconvexities in this problem arise due to the existence of polynomial terms in either the objective function or the set of constraints. We assume here that the polynomials have nonconvex terms up to the Nth degree term.

3.10.0.1 Primal Problem and The Lagrange Function. Consider the following transformations:

$$\begin{array}{rcccl} x_0 & = & 1 & & \\ x_1 & = & y & & \\ x_2 & = & y^2 & = & (x_1)y \\ & \vdots & & \vdots & \\ x_N & = & y^N & = & (x_{N-1})y. \end{array} \tag{3.35}$$

Then, problem (3.34) can be written in the following equivalent form:

$$\min_{\boldsymbol{x},y} \sum_{i=0}^{N} a_i x_i$$

$$\begin{array}{rcll} \sum_{i=0}^{N} A_{ji} x_i & \leq & 0 & j = 1, 2, \ldots, J \\ \sum_{i=0}^{N} B_{mi} x_i & = & 0 & m = 1, 2, \ldots, M \\ x_i - x_{i-1} y & = & 0 & i = 1, 2, \ldots, N \end{array} \tag{3.36}$$

where $x_0 = 1$.

Since the cases of unconstrained and constrained problems differ in both the solution of the primal and the development of the Lagrange function, they are considered separately here.

Unconstrained Problems In the absence of constraints, problem (3.36) can be simplified to

$$\begin{array}{rrcll} & \min_{\boldsymbol{x},y} & & \sum_{i=0}^{N} a_i x_i & \\ s.t. & x_i - x_{i-1} y & = & 0 & i = 1, 2, \ldots, N \\ & x_0 & = & 1. & \end{array} \tag{3.37}$$

For a fixed $y = y^K$, the solution of the primal problem can be found as follows:

$$\begin{array}{rcl} x_0 & = & 1 \\ x_1 & = & y^K \\ x_2 & = & (y^K)^2 \\ & \vdots & \\ x_N & = & (y^K)^N \\ f(y^K) & = & \sum_{i=0}^{N} a_i (y^K)^i. \end{array}$$

The KKT gradient conditions for the primal problem can be written as

$$\nabla_{x_i} L(\boldsymbol{x}, y^K, \nu^K) = a_i + \nu_i^K - \nu_{i+1}^K y^K = 0 \quad \forall\, i = 1, 2, ..., N$$

where ν_i^K are the Lagrange multipliers for the new equality constraints introduced, with $\nu_{N+1}^K = 0$. Thus, the Lagrange multipliers are found by backward substitution:

$$\begin{array}{rcl}
\nu_N^K & = & -a_N \\
\nu_{N-1}^K & = & \nu_N^K y^K - a_{N-1} \quad = \quad -a_N(y^K) - a_{N-1} \\
\nu_{N-2}^K & = & \nu_{N-1}^K y^K - a_{N-2} \quad = \quad -a_N(y^K)^2 - a_{N-1}(y^K) - a_{N-2} \\
\vdots & & \vdots \qquad\qquad\qquad\qquad \vdots \\
\nu_0^K & = & \nu_1^K y^K - a_0 \quad = \quad -a_N(y^K)^N \cdots - a_1(y^K) - a_0.
\end{array}$$

The Lagrange function formulated from the primal problem can be written as

$$\begin{aligned}
L(\boldsymbol{x}, y, \nu^K) &= \sum_{i=0}^{N} a_i x_i + \sum_{i=1}^{N} \nu_i^K (x_i - x_{i-1} y) + \nu_0^K (x_0 - 1) \\
&= \sum_{i=0}^{N} \left[a_i + \nu_i^K - \nu_{i+1}^K y\right] x_i - \nu_0^K. \qquad (3.38)
\end{aligned}$$

Using the KKT gradient conditions, the Lagrange function becomes

$$L(\boldsymbol{x}, y, \nu^K) = \sum_{i=0}^{N} \nu_{i+1}^K (y^K - y) x_i - \nu_0^K. \qquad (3.39)$$

Constrained Problems When the optimization problem involves constraints, the primal problem can no longer be solved by function evaluation. In these cases, the primal problem, for a fixed $y = y^K$, can be written as

$$\min_{\boldsymbol{x}} \sum_{i=0}^{N} a_i x_i$$

$$\begin{aligned}
\sum_{i=0}^{N} A_{ji} x_i &\leq 0 \qquad j = 1, 2, ..., J \\
\sum_{i=0}^{N} B_{mi} x_i &= 0 \qquad m = 1, 2, ..., M \qquad (3.40) \\
x_i - x_{i-1} y^K &= 0 \qquad i = 1, 2, ..., N
\end{aligned}$$

where $x_0 = 1$. The solution of this problem is obtained by solving a linear programming problem in the $\boldsymbol{x}$ variables, and provides the multipliers used in formulating the Lagrange function.

The KKT gradient conditions for the primal problem are

$$\begin{aligned}\nabla_{x_i} L(\boldsymbol{x}, y^K, \lambda^K, \mu^K, \nu^K) &= a_i + \sum_{j=1}^{J} \mu_j^K A_{ji} + \sum_{m=1}^{M} \lambda_m^K B_{mi} \\ &+ \nu_i^K - \nu_{i+1}^K y^K = 0 \end{aligned} \tag{3.41}$$

where λ^K and μ^K correspond to the original equality and inequality constraints, and ν^K corresponds to the new equivalence constraints, with $\nu_{N+1}^K = 0$.

The Lagrange function for this problem is given by

$$\begin{aligned} L(\boldsymbol{x}, y, \lambda^K, \mu^K, \nu^K) &= \sum_{i=0}^{N} \left[a_i x_i + \sum_{j=1}^{J} \mu_j^K A_{ji} x_i \right] \\ &+ \sum_{i=0}^{N} \left[\sum_{m=1}^{M} \lambda_m^K B_{mi} x_i + \nu_i^K (x_i - x_{i-1} y) \right] \\ &+ \nu_0^K (x_0 - 1) \\ &= \sum_{i=0}^{N} \left[a_i + \sum_{j=1}^{J} \mu_j^K A_{ji} \right] x_i \\ &+ \sum_{i=0}^{N} \left[\sum_{m=1}^{M} \lambda_m^K B_{mi} + \nu_i^K - \nu_{i+1}^K y \right] x_i - \nu_0^K . \end{aligned} \tag{3.42}$$

Using the KKT conditions, it can be shown easily that the Lagrange function again reduces to the form (3.39).

Infeasible Primal Problems Due to the presence of constraints, it is possible that for some values of $y = y^K$, the primal problem is infeasible. In this case, it is necessary to solve a relaxed primal problem involving the minimization of the sum of infeasibilities. This problem is shown below:

$$\min_{\boldsymbol{x}} \sum_{j=1}^{J} \alpha_j + \sum_{m=1}^{M} (\beta_m + \gamma_m)$$

$$\begin{aligned} \sum_{i=0}^{N} A_{ji} x_i &\leq \alpha_j & j &= 1, 2, \ldots, J, \\ \sum_{i=0}^{N} B_{mi} x_i &= \beta_m - \gamma_m & m &= 1, 2, \ldots, M, \\ x_i - x_{i-1} y^K &= 0 & i &= 1, 2, \ldots, N . \end{aligned}$$

The Lagrange function is given by

$$\begin{aligned} L(\boldsymbol{x}, y, \lambda^K, \mu^K, \nu^K) &= \sum_{j=1}^{J} \alpha_j + \sum_{m=1}^{M} (\beta_m + \gamma_m) + \sum_{j=1}^{J} \mu_j^K \sum_{i=1}^{N} A_{ji} x_i - \alpha_j \\ &+ \sum_{m=1}^{M} \lambda_m^K (\sum_{i=1}^{N} B_{mi} x_i - \beta_m + \gamma_m) \\ &+ \sum_{i=1}^{N} \nu_i^K (x_i - x_{i-1} y) \ . \end{aligned} \tag{3.43}$$

Using the KKT gradient conditions for the relaxed primal problem and separating the terms in $\boldsymbol{x}$, it can be easily shown that this again reduces to the form (3.39).

3.10.0.2 Relaxed Dual Problems. As shown in the previous section, for the cases of unconstrained as well as constrained problems, the Lagrange function is of the form:

$$L(\boldsymbol{x}, y, \nu^K) = \sum_{i=0}^{N} \nu_{i+1}^K (y^K - y) x_i - \nu_0^K .$$

Thus, the *connected* variables are $x_1, x_2, \ldots, x_N$. However, the *qualifying* constraint for each x_i (i.e., the gradient of the Lagrange function w.r.t. that x_i) is always of the form

$$y^K - y \geq 0, \ or \quad y^K - y \leq 0,$$

depending on whether ν_{i+1}^k is greater than or less than zero respectively. This implies that only two relaxed dual problem problems (one for $y^K - y \geq 0$ and one for $y^K - y \leq 0$) need to be solved at each iteration. To get the underestimators of the Lagrange function for these two regions, the bounds for x_i can be found as follows:

(i) For $y^K - y \geq 0$,

$$x_i^B = \begin{cases} x_i^L \text{ if } \lambda_{i+1}^K \geq 0 \\ x_i^U \text{ if } \lambda_{i+1}^K < 0 \end{cases}$$

(ii) For $y^K - y \leq 0$,

$$x_i^B = \begin{cases} x_i^U \text{ if } \lambda_{i+1}^K \geq 0 \\ x_i^L \text{ if } \lambda_{i+1}^K < 0. \end{cases}$$

These two problems are separately considered below.

First Relaxed Dual Problem Consider the relaxed dual problem solved for the region $y^K - y \geq 0$. Before solving the relaxed dual problems, the Lagrange functions from previous iterations are selected. From the kth iteration ($k = 1, 2, ..., K-1$), the two Lagrange functions have *qualifying* constraints of the form $y^k - y \geq 0$ and $y^k - y \leq 0$. These Lagrange functions are selected on the basis of satisfaction of their *qualifying* constraints at $y = y^K$. Since $y^k - y^K$ must be either positive or negative, exactly one Lagrange function from every iteration will be present for the current relaxed dual problems.

Now, the previous iterations correspond to values of $y = y^1, \ldots, y^{K-1}$. Some of these fixed values of y will be less than y^K. Suppose that

$$y^L = \max_{j=1,\ldots,K-1} \{y^j : y^j < y^K\}.$$

Then, the Lagrange function formulated from the Lth iteration (for which the fixed value of y is y^L) has the *qualifying* constraint $y^L - y \leq 0$. Therefore, for the current relaxed dual problem, the lower bound on the y variable is y^L. At the same time, the *qualifying* constraint from the current iteration ensures that the upper bound on y is y^K. Thus, for the current relaxed dual problem, $y^L \leq y \leq y^K$.

Now, the x variables are related to y through the equivalence relations (3.35). Therefore, the bounds on the x variables can be changed to suit the new bounds on y for the current relaxed dual problem. This is done in the following manner:

$$\begin{array}{rclrcl}
x_1^L & = & y^L\,, & x_1^U & = & y^K \\
x_2^L & = & MIN[(y^L)^2, (y^K)^2]\,, & x_2^U & = & MAX[(y^L)^2, (y^K)^2] \\
x_3^L & = & MIN[(y^L)^3, (y^K)^3]\,, & x_3^U & = & MAX[(y^L)^3, (y^K)^3] \\
\vdots & & \vdots & \vdots & & \vdots \\
x_N^L & = & MIN[(y^L)^N, (y^K)^N]\,, & x_N^U & = & MAX[(y^L)^N, (y^K)^N].
\end{array}$$

If $y^L < 0$ and $y^K > 0$, then $x_2^L = x_4^L = x_6^L \ldots = 0$.

The reason for the use of these expressions is as follows. If $y^L \geq 0$, then the lower and upper bounds on $x_1, x_2, \ldots x_N$ are simply given by

$$x^L = (y^L, (y^L)^2, \ldots (y^L)^N), \text{and} \tag{3.44}$$

$$x^U = (y^K, (y^K)^2, \ldots (y^K)^N). \tag{3.45}$$

However, if $y^L < 0$ and $y^K > 0$, then the value of y can be either positive or negative. Therefore, any even power of y can be as low as zero. Hence, $x_2^L = x_4^L \ldots = 0$. Furthermore, $\mid y^L \mid$ can be either greater than or less than $\mid y^K \mid$. Therefore, for each of the powers of y, the minimum and maximum values are given by the minimum and maximum values of the corresponding terms in *both* y^L and y^K.

These new bounds for the x variables are then used in the Lagrange functions in the following form:

$$x_i^B = \begin{cases} x_i^L & \text{if } \nu_{i+1} \geq 0 \\ x_i^U & \text{if } \nu_{i+1} \leq 0. \end{cases}$$

It should be noted here that due to the nature of the transformation variables that are introduced into the problem, the Lagrange function is linear in x for every fixed y.

There are $(K-1)$ Lagrange functions from previous iterations that can be used for the current relaxed dual problem. However, it is sufficient to consider only one of these Lagrange functions, since the omission of the remaining constraints does not destroy the validity of the relaxed dual problem as a lower bound on the global solution. The obvious choice is to use the Lagrange function corresponding to the nearest point on the left side, that is, the Lagrange function corresponding to y^L, and ignoring all other Lagrange functions from previous iterations. The current relaxed dual problem then has the following form:

$$\min_{y,\mu_B} \mu_B$$

$$\begin{aligned} s.t. & \\ \mu_B \geq \; & L(x^{B_L}, y, \nu^L) \\ & y \geq y^L \\ \mu_B \geq \; & L(x^{B_K}, y, \nu^K) \\ & y \leq y^K \end{aligned} \tag{3.46}$$

where B_L and B_K are two combinations of bounds of the x variables that are being used in the Lagrange functions from the Lth and Kth iterations, respectively.

Due to the nature of the transformations, the two constraints are linear in y. Therefore, it is clear that the solution to this problem will be either at the intersection of the two constraints, or at one of the two bounds for y, namely, y^L or y^K. Consider the three cases.

(i) The solution of the current relaxed dual problem lies at y^L. In this case, the value of the first Lagrange function (that is, the one formulated from y^L) must equal the value of the objective function at y^L. This arises as a result of the strong duality theorem. This means that due to the presence of the first constraint, the value of μ_B must equal the value of the objective function at y^L. Therefore, it is not necessary to consider this solution.

(ii) The solution of the current relaxed dual problem lies at y^K. In this case, the value of the second Lagrange function (that is, the one formulated from y^K) must equal the value of the objective function at y^K. This arises as a result of the strong duality theorem. This means that due to the presence of the second constraint, the value of μ_B must equal the value of the objective function at y^K. Therefore, it is not necessary to consider this solution.

(iii) The solution of the current relaxed dual problem lies at the intersection of the two constraints. In this case, the solution needs to be determined.

Thus, the relaxed dual problem can be solved easily by considering only the third possibility and solving for the intersection of the two constraints in the problem. This is very easy to do since the constraints are linear in y.

Second Relaxed Dual Problem For the case when $y^K - y \leq 0$, the nearest point to the right side of y^K becomes the upper bound for y, while the value of y^K becomes the lower bound for y. That is , for this case, $y^K \leq y \leq y^R$, where now y^R is found as

$$y^R = \min_{j=1,\ldots,K-1} \{y^j : y^j > y^K\} \ .$$

Similarly, the bounds for the x variables can be found as follows:

$$\begin{array}{rclrcl}
x_1^L & = & y^K \ , & x_1^U & = & y^R \\
x_2^L & = & MIN[(y^K)^2, (y^R)^2] \ , & x_2^U & = & MAX[(y^K)^2, (y^R)^2] \\
x_3^L & = & MIN[(y^K)^3, (y^R)^3] \ , & x_3^U & = & MAX[(y^K)^3, (y^R)^3] \\
\vdots & & \vdots & \vdots & & \vdots \\
x_N^L & = & MIN[(y^K)^N, (y^R)^N] \ , & x_N^U & = & MAX[(y^K)^N, (y^R)^N] \ .
\end{array}$$

If $y^K < 0$, then $x_2^L = x_4^L = x_6^L \ldots = 0$.

In this case, the relaxed dual problem is solved by considering the current Lagrange function, that is, the Lagrange function formulated from the current iteration corresponding to $y^K - y \leq 0$, and the Lagrange function from the Rth iteration corresponding to $y^R - y \geq 0$. Again, the relaxed dual problem can be solved simply by considering the intersection of the two Lagrange functions and comparing the corresponding value of μ_B to the values of the objective function at y^K and y^R.

The GOP/POLY Algorithm

The modified algorithm for minimizing unconstrained and constrained polynomial functions in one variable is given in Visweswaran and Floudas (1992). The key point to note about this specific version is that the primal problem for unconstrained problems involves only a function evaluation, and the two relaxed dual subproblems are solved very simply by calculation of the intersection of two linear equality constraints in two variables. This results in significant improvements in the time needed by the algorithm to find and converge to a global solution.

3.11 ADDITIONAL PROPERTIES OF THE LAGRANGE FUNCTION

Application of the GOP algorithm led to the observation that a large number of the relaxed dual subproblems are either infeasible or have identical solutions. This suggested that by looking more carefully into the structure of the Lagrange function, it might be possible to avoid solving some of these subproblems (see Visweswaran and Floudas (1993)). To exploit further the mathematical structure we introduce the following condition, denoted as *Condition (A')* :

At each iteration, the Lagrange function (or its linearization) that is used in the relaxed dual *problem is linear in $\boldsymbol{x}$ for every fixed $\boldsymbol{y}$, and linear in $\boldsymbol{y}$ for every fixed $\boldsymbol{x}$.*

At first glance, *Condition (A')* seems quite restrictive. However, this is not really the case. By definition, the Lagrange function at iteration k (given by (3.6)) is a linear combination of the objective function and constraints. Thus, it can be seen that any problem where $f(\boldsymbol{x},\boldsymbol{y})$, $g(\boldsymbol{x},\boldsymbol{y})$ and $h(\boldsymbol{x},\boldsymbol{y})$ are linear in $\boldsymbol{x}$ for fixed $\boldsymbol{y}$ and vice-versa will involve Lagrange functions that automatically satisfy *Condition (A')*. Therefore, through an appropriate introduction of *transformation* variables and *partitioning* of the resulting variable set, it can be shown that *Condition (A')* is satisfied by problems consisting of quadratic or polynomial terms in the objective function and/or constraints.

Condition (A') implies that the linearization of the Lagrange function with respect to $\boldsymbol{x}$ is the same as the Lagrange function. In other words,

$$L(\boldsymbol{x},\boldsymbol{y},\lambda^k,\mu^k)\big|_{\boldsymbol{x}^k}^{lin} = L(\boldsymbol{x},\boldsymbol{y},\lambda^k,\mu^k), \text{ and}$$

$$g_i^k(\boldsymbol{y}) = \nabla_{x_i} L(\boldsymbol{x},\boldsymbol{y},\lambda^k,\mu^k)\big|_{\boldsymbol{x}^k}^{lin} = \nabla_{x_i} L(\boldsymbol{x},\boldsymbol{y},\lambda^k,\mu^k) \qquad \forall\, i.$$

Moreover, this means that $g_i^k(\boldsymbol{y})$ are linear functions of $\boldsymbol{y}$. Therefore, using interval analysis, it is easy to determine lower and upper bounds for $g_i^k(\boldsymbol{y})$ over Y. The values of these bounds are useful in restricting the number of relaxed dual subproblems that need to be solved at iteration k, as stated in the following property.

Property 3.11.1 *At the kth iteration of the GOP algorithm, if $g_j^k(\boldsymbol{y}) \geq 0$ for all $\boldsymbol{y}$ (respectively $g_j^k(\boldsymbol{y}) \leq 0$ for all y), it is not necessary to solve those* relaxed dual *subproblems that have $g_j^k(\boldsymbol{y}) \leq 0$ (respectively $g_j^k(\boldsymbol{y}) \geq 0$) as a* qualifying *constraint.*

Proof. The Lagrange function formulated from the kth primal problem has the following form :

$$\begin{aligned} L(\boldsymbol{x},\boldsymbol{y},\lambda^k,\mu^k) &= L(\boldsymbol{x}^k,\boldsymbol{y},\lambda^k,\mu^k) + g^k(\boldsymbol{y})\cdot(\boldsymbol{x}-\boldsymbol{x}^k) \\ &= L(\boldsymbol{x}^k,\boldsymbol{y},\lambda^k,\mu^k) + \sum_{i\neq j} g_i^k(\boldsymbol{y})(x_i - x_i^k) + g_j^k(\boldsymbol{y})(x_j - x_j^k). \end{aligned}$$

Consider the two *relaxed dual* subproblems that are solved at the kth iteration for two combinations of bounds of the *connected* $\boldsymbol{x}$ variables which differ only in the bound of x_j. That is, consider the following two combinations B_1 and B_2 :

$$B_1 \equiv \{x_1^B, x_2^B, \ldots, x_{j-1}^B, x_j^L, x_{j+1}^B, \ldots, x_n^B\}\,, \quad \text{and} \tag{3.47}$$

$$B_2 \equiv \{x_1^B, x_2^B, \ldots, x_{j-1}^B, x_j^U, x_{j+1}^B, \ldots, x_n^B\}\,. \tag{3.48}$$

The two corresponding *relaxed dual* subproblems are shown below :

(i) For $\boldsymbol{x}^B = \boldsymbol{x}^{B_1}$

$$\begin{aligned} & \min_{\boldsymbol{y},\mu_B} \ \mu_B \\ s.t. & \\ \mu_B \ &\geq \ L(\boldsymbol{x}^k, \boldsymbol{y}, \lambda^k, \mu^k) + \sum_{i \neq j} g_i^k(\boldsymbol{y})(x_i^B - x_i^k) + g_j^k(\boldsymbol{y})(x_j^L - x_j^k) \\ & g_j^k(\boldsymbol{y}) \geq 0 \\ F(\mu_B, \boldsymbol{y}) \ &\leq \ \mathbf{0}. \end{aligned} \tag{3.49}$$

(ii) For $\boldsymbol{x}^B = \boldsymbol{x}^{B_2}$

$$\begin{aligned} & \min_{\boldsymbol{y},\mu_B} \ \mu_B \\ s.t. & \\ \mu_B \ &\geq \ L(\boldsymbol{x}^k, \boldsymbol{y}, \lambda^k, \mu^k) + \sum_{i \neq j} g_i^k(\boldsymbol{y})(x_i^B - x_i^k) + g_j^k(\boldsymbol{y})(x_j^U - x_j^k) \\ & g_j^k(\boldsymbol{y}) \leq 0 \\ F(\mu_B, \boldsymbol{y}) \ &\leq \ \mathbf{0}. \end{aligned} \tag{3.50}$$

In both problems, the last constraint ($F(\mu_B, \boldsymbol{y}) \leq 0$) has been added to represent all other constraints (such as Lagrange functions and their *qualifying* constraints from previous iterations, constraints from the original problem etc.) that are common to all the *relaxed dual* subproblems.

Consider the *qualifying* constraint corresponding to the variable x_j for these two subproblems. The function $g_j^k(\boldsymbol{y})$ is linear in $\boldsymbol{y}$. Hence, assuming that the bounds on the $\boldsymbol{y}$ variables are known, it is possible to determine whether $g_j^k(\boldsymbol{y})$ is at one of its bounds (lower or upper). Suppose that such a check reveals that $g_j^k(\boldsymbol{y}) \geq 0$ is always true. Then, this can be added as a constraint to both problems (3.49) and (3.50). This does not affect problem (3.49). However, problem (3.50) becomes

$$\begin{aligned} & \min_{\boldsymbol{y},\mu_B} \ \mu_B \\ s.t. & \\ \mu_B \ &\geq \ L(\boldsymbol{x}^k, \boldsymbol{y}, \lambda^k, \mu^k) + \sum_{i \neq j} g_i^k(\boldsymbol{y})(x_i^B - x_i^k) + g_j^k(\boldsymbol{y})(x_j^U - x_j^k) \\ & g_j^k(\boldsymbol{y}) \leq 0 \\ F(\mu_B, \boldsymbol{y}) \ &\leq \ \mathbf{0} \\ & g_j^k(\boldsymbol{y}) \geq 0. \end{aligned} \tag{3.51}$$

From this, it can be seen that for problem (3.50), the addition of the new constraint is equivalent to setting $g_j^k(\boldsymbol{y}) = 0$. Hence, the first constraint in

problem (3.51) becomes

$$\begin{aligned}\mu_B &\geq L(\boldsymbol{x}^k, \boldsymbol{y}, \lambda^k, \mu^k) + \sum_{i \neq j} g_i^k(\boldsymbol{y})(x_i^B - x_i^k) + g_j^k(\boldsymbol{y}).(x_j^U - x_j^k) \\ &\geq L(\boldsymbol{x}^k, \boldsymbol{y}, \lambda^k, \mu^k) + \sum_{i \neq j} g_i^k(\boldsymbol{y})(x_i^B - x_i^k). \end{aligned} \tag{3.52}$$

Hence, problem (3.50) can be written as

$$\begin{aligned} &\min_{y, \mu_B} \ \mu_B \\ s.t. \\ \mu_B &\geq L(\boldsymbol{x}^k, \boldsymbol{y}, \lambda^k, \mu^k) + \sum_{i \neq j} g_i^k(\boldsymbol{y})(x_i^B - x_i^k) \\ &g_j^k(\boldsymbol{y}) = 0 \\ F(\mu_B, \boldsymbol{y}) &\leq \mathbf{0}. \end{aligned} \tag{3.53}$$

At the same time, problem (3.49) can be written as

$$\begin{aligned} &\min_{y, \mu_B} \ \mu_B \\ s.t. \\ \mu_B &\geq L(\boldsymbol{x}^k, \boldsymbol{y}, \lambda^k, \mu^k) + \sum_{i \neq j} g_i^k(\boldsymbol{y})(x_i^B - x_i^k) + g_i^k(\boldsymbol{y})(x_j^L - x_j^k) \\ &g_j^k(\boldsymbol{y}) \geq 0 \\ F(\mu_B, \boldsymbol{y}) &\leq \mathbf{0}. \end{aligned} \tag{3.54}$$

Since $g_j^k(\boldsymbol{y}) \geq 0$ and $x_j^L \leq x_j^k$, the solution of problem (3.54) is always less than or equal to the solution of problem (3.53). Also, the feasible region of problem (3.54) contains the feasible region of problem (3.53). Hence, it is not necessary to solve the problem (3.53) (since, according to the GOP algorithm, we select the minimum of the solutions of the *relaxed dual* subproblems solved at all the combinations of bounds of the *connected* x variables). Moreover, since the set of bounds for the $x_i, i \in \{1, 2..n\} \backslash \{j\}$ was chosen arbitrarily, it is not necessary to solve the relaxed dual problem for any combination of bounds of the $\boldsymbol{x}$ variables for which $g_j^k(\boldsymbol{y}) \leq 0$ is a *qualifying* constraint. Similarly, the opposite case (when $g_j^k(\boldsymbol{y})$ is always negative) can be proved.

Figure 3.6 shows a graphical illustration of this property. For this case, there are two $\boldsymbol{x}$ and two $\boldsymbol{y}$ variables. Normally, there would be one *relaxed dual* subproblem solved in each of regions 1, 2, 3 and 4. However, it can be seen that y^1 lies on the boundary. This is mathematically indicated by the fact that inside the feasible region (the complete rectangle) $g_2^1(\boldsymbol{y})$ is always positive. Therefore, this property indicates that regions 1 and 4 will not provide any useful solutions, and the relaxed dual subproblems for those regions can be omitted.

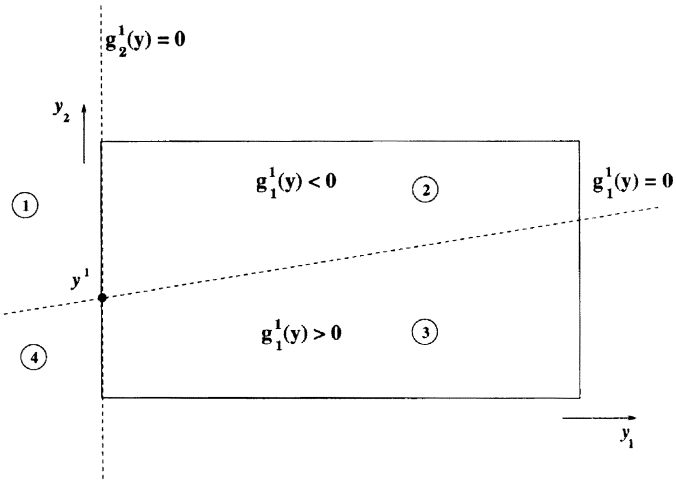


Figure 3.6. Illustration of Property 3.11.1

It is also possible that at any iteration, some of the gradients vanish. In such cases, again, it is possible to reduce the number of relaxed dual subproblems, as shown by the following property.

Property 3.11.2 *At iteration k of the GOP algorithm, if $g_j^k(\boldsymbol{y}) = 0$ for all $\boldsymbol{y}$, then, it is sufficient to solve for only those combinations of bounds for which x_j is at either its lower or upper bound.*

Proof. Since $g_j^k(\boldsymbol{y}) = 0$, the Lagrange functions formulated from the kth iteration and used in the two *relaxed dual* problems (3.49) and (3.50) are identical. Also, the *qualifying* constraint w.r.t x_j for the two problems are the same, being of the form $g_j^k(\boldsymbol{y}) = 0$. Hence, problems (3.49) and (3.50) have the same solution. Therefore, it is sufficient to solve either problem (3.49) or problem (3.50). Since this holds regardless of the set of bounds for the $x_i, i \in 1, 2..n \backslash \{j\}$, it is sufficient to solve for only those combinations of bounds that have x_j set to either its upper or its lower bound.

As an illustration, suppose that the gradient of the Lagrange function *w.r.t.* x_2 in the previous example is zero. Then, in Figure 3.6, there would be no need to partition the region with $g_2^1(\boldsymbol{y}) = 0$. Rather, only two relaxed dual subproblems would need to be solved, one for the composite region 1+2 and one for the region 3+4. Thus, as with Property 3.11.1, the result is a reduction in the number of relaxed dual subproblems solved by a factor of 2, for every x variable for which the gradient vanishes.

For several classes of problems, the $\boldsymbol{x}$ and $\boldsymbol{y}$ variable sets are coupled. For example, for the case of quadratic problems, the $\boldsymbol{y}$ variable set does not originally

exist; it is introduced into the problem through the use of *transformations*. In such cases, there is a one-to-one correspondence between the $\boldsymbol{x}$ and $\boldsymbol{y}$ variable sets. The following property shows how to exploit this feature.

Property 3.11.3 *Suppose that there is a one-to-one correspondence between the $\boldsymbol{x}$ and $\boldsymbol{y}$ variable set in the feasible region of the problem, and that $y_i = x_i$. Then, the bounds on the* connected *$\boldsymbol{x}$ variables can be updated without destroying the property of ϵ-global optimality of the GOP algorithm.*

Proof. By the assumption of one-to-one correspondence, there is a y_i for every x_i. Moreover, it is assumed that $y_i = x_i$. Then, consider the Kth iteration of the GOP algorithm. The fixed value of $\boldsymbol{y}$ for the primal problem is $\boldsymbol{y}^K$. Then, before solving the *relaxed dual* subproblems for the current (Kth) iteration, the Lagrange functions from previous iterations are selected to be present in the *relaxed dual* subproblems for the current (Kth) iteration. Since these Lagrange functions are selected on the basis of their *qualifying* constraints being satisfied at $\boldsymbol{y}^K$, there will be one Lagrange function from every iteration present in the *relaxed dual* subproblems of the Kth iteration. Along with this Lagrange function, there will be a *qualifying* constraint of the form $g_i^k(\boldsymbol{y}) \geq 0$ or $g_i^k(\boldsymbol{y}) \leq 0$.

Let $y^1, y^2 ... y^{K-1}$ represent the fixed values of y for all iterations prior to the Kth iteration. Then, for $y_i, i = 1, 2 ... n$, it is possible to determine the lower and upper bounds, that is, for every i,

$$y_i^L = MIN(y_i^1, y_i^2 ... y_i^{K-1}) \quad \text{and} \quad y_i^U = MAX(y_i^1, y_i^2 ... y_i^{K-1}) \quad (3.55)$$

can be determined. Then, it follows that $y_i^K \in [y_i^L, y_i^U] \quad \forall\, i$.

The Lagrange functions that are added from the previous iterations have their *qualifying* constraints satisfied at $\boldsymbol{y}^K$. Since these constraints are linear in $\boldsymbol{y}$, and are satisfied identically as an equality at the respective $\boldsymbol{y}^k$, it follows that they are satisfied for all $\boldsymbol{y}$ between $\boldsymbol{y}^k$ and $\boldsymbol{y}^K$. Moreover, the presence of these *qualifying* constraints in the *relaxed dual* subproblems ensures that the values of y_i are restricted to lie between y_i^L and y_i^U. Since there is a direct correspondence between the values of $\boldsymbol{y}$ and $\boldsymbol{x}$, this means that for every i, the bounds of x_i can be updated to (x_i^L, x_i^U).

Property 3.11.3 can be illustrated by the same example as before. However, suppose now that $x_1 = y_1$ and $y_2 = x_2$. Then, consider a case as in Figure 3.7, which shows the 4th iteration of the algorithm. Here, y^1, y^2, y^3 and y^4 are the fixed values of $\boldsymbol{y}$ for the first four primal problems. It can be seen that when the *relaxed dual* subproblem for region 1 is solved, y_1 must lie between y_1^2 and y_1^4. Because of the one-to-one correspondence between $\boldsymbol{x}$ and $\boldsymbol{y}$, this means that the bounds used for the $\boldsymbol{x}$ variables for this problem should be (y_1^2, y_2^4) for x_1 and (y_2^2, y_2^4) for x_2 (rather than the original lower and upper bounds, which are indicated by the complete rectangle.) This leads to a better underestimation of the optimal solution in region 1 resulting in a tighter (but still rigorous) lower bound. Similar explanations hold for the other three regions. Therefore, when this property is applicable, faster convergence of the algorithm is achieved.

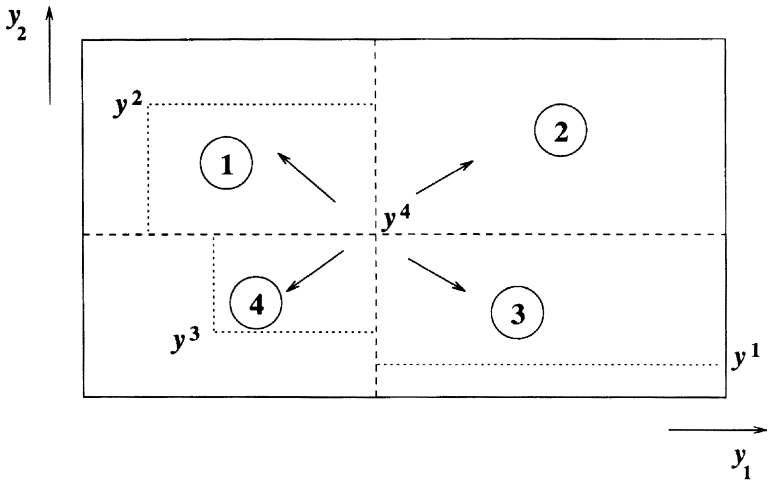


Figure 3.7. Illustration of Property 3.11.3

3.11.0.3 Iterations with Infeasible Primal Problems. Often, the values of y^{k+1} that are returned by Step 4 of the GOP algorithm after solving the relaxed dual subproblems and selecting the minimum solution are such that the primal problem is infeasible. Suppose that this happens at the kth iteration, that is, the primal problem, when solved for $y = y^k$, does not have any feasible solution. Then, it is necessary to solve a *relaxed primal* problem for this fixed value of $y = y^k$. Because of the way in which the *relaxed primal* problem is solved, the Lagrange functions formulated from this problem (having the form $L_1(x, y, \lambda^k, \mu^k) \leq 0$) are used mainly as constraints defining the feasible region for the relaxed dual subproblems in terms of y. At the same time, if there are other stored solutions remaining (from the solutions of the relaxed dual subproblems of previous iterations), it is likely that some of these solutions provide values of y that are feasible for the primal problem. Therefore, for iterations when the primal problem is infeasible (such as the kth iteration), the solution of the *relaxed primal* problem is used to generate constraints to be used for future relaxed dual subproblems, but to not actually solve the relaxed dual subproblems for the current (kth) iteration. This strategy can be applied in general as long as there are stored solutions remaining at the kth iteration, since it is always possible to backtrack to the kth iteration if no feasible solutions can be found by future iterations.

3.11.1 Illustrative Example : A Multiperiod Tankage Quality Problem

This example concerns a multiperiod tankage quality problem that arises often in the operations of refineries. The problem concerns the optimal allocation of

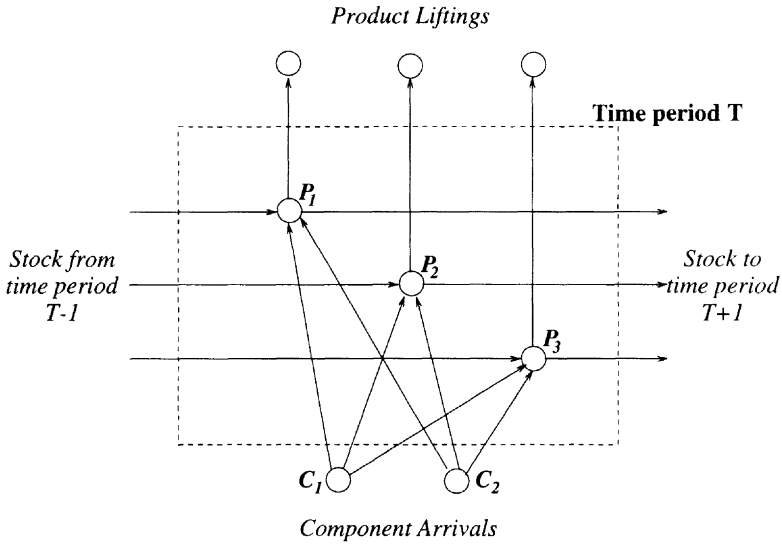


Figure 3.8. Multiperiod Tankage Quality Problem

a set of components used in blending products of different grades. In order to develop the mathematical formulation, the following sets are defined :

$$
\begin{aligned}
PR &= \{p\} \equiv \text{set of products} \\
CO &= \{c\} \equiv \text{set of components} \\
T &= \{t\} \equiv \text{set of time periods} \\
QL &= \{l\} \equiv \text{set of qualities}
\end{aligned}
$$

For this problem, there are 3 products (p_1, p_2, p_3), 2 components (c_1, c_2), and 3 time periods (t_0, t_1, t_2). The following variables are defined :

$$
\begin{aligned}
x_{c,p,t} &= \text{amount of component } c \text{ allocated to product } p \text{ at period } t \\
s_{p,t} &= \text{stock of product } p \text{ at end of period } t \\
q_{p,l,t} &= \text{quality } l \text{ of product } p \text{ at period } t
\end{aligned}
$$

A typical time period is shown in Figure 3.8.

The objective of the problem is to maximize the total value at the end of the last time period. The terminal value of each product (v_p) is given. Lower and upper bounds on the qualities of the products, qualities of stocks at the start of each time period $(s_{p,t})$, qualities in each component $(QU_{c,l})$, and the product lifting $(LF_{p,t})$ for every period are provided. The data for this problem is shown in Table 3.1.

Component Arrivals and Qualities

Component	*Arrivals*			*Qualities*	
	t_0	t_1	t_2	q_1	q_2
c_1	0.20	0.25	0.15	40	80
c_2	0.20	0.15	0.25	100	50

Product Lifting and Limits on Stocks

Product	*Product Lifting*		*Stock Limits*		
	t_1	t_2	t_0	t_1	t_2
p_1	0.08	0.12	0.05	0.10	0.10
p_2	0.15	0.10	0.05	0.10	0.10
p_3	0.15	0.20	0.05	0.10	0.10

Bounds and Initial Values for Product Qualities

Products	*Lower Bounds*		*Upper Bounds*		*Initial Values*	
	q_1	q_2	q_1	q_2	q_1	q_2
p_1	70	50	100	100	70	50
p_2	80	70	100	100	90	70
p_3	60	40	100	100	60	40

Terminal Value of products : $v_p = (60, 90, 40)$.

Table 3.1. Data for the Multiperiod Tankage Quality Problem

The complete mathematical formulation for this problem, consisting of 39 variables and 22 inequality constraints (of which 12 are nonconvex) is given below :

$$
\begin{array}{rcll}
 & & \max & \displaystyle\sum_{p \in PR} v_p . s_{p,'t'_2} \\
s.t. & & & \\
 & & \displaystyle\sum_{p \in PR} x_{c,p,t} \leq AR_{c,t} & t \in \{t_1, t_2\},\ c \in CO \\
s_{p,t} + \displaystyle\sum_{c \in CO} x_{c,p,t+1} - s_{p,t+1} & \geq & LF_{p,t+1} & t \in \{t_0, t_1\},\ p \in PR \\
s_{p,t} . q_{p,l,t} + \displaystyle\sum_{c \in CO} x_{c,p,t+1} . QU_{c,l} & \geq & & \\
 & & (s_{p,t+1} + LF_{p,t+1}) \cdot q_{p,l,t+1} & t \in \{t_0, t_1\},\ p \in PR,\ l \in QL.
\end{array}
$$

The sources of nonconvexities in this problem are the bilinear terms $s_{p,t} \cdot q_{p,l,t}$ in the last set of constraints. Thus, fixing either the set of s or q variables makes the problem linear in the remaining variables.

3.11.1.1 The GOS Algorithm. This problem was first studied through the application of the Global Optimum Search (GOS) approach (Floudas *et al.*, 1989; Floudas and Aggarwal, 1990). The GOS was successful in identifying the global minimum for the problem, however from some of the starting points, it converged to a local solution, despite the application of a "Radial Search" feature.

3.11.1.2 The GOP Algorithm. Using the GOP algorithm to solve this problem, we *project* on the qualities (q_1, q_2). Then, the stocks are the *connected* variables. Since there are six of them (corresponding to three products at two time periods), 64 relaxed dual subproblems need to be solved at every iteration.

When the original GOP algorithm (without the additional properties given in this section) was applied, it converged to the global solution of -9.5316 from all considered starting points. Starting from the lower bound, the algorithm took 16 iterations to converge, solving a total of 16 primal and 1024 relaxed dual subproblems. In addition, the algorithm needed to solve 3 *relaxed primal* problems.

When we use Properties 3.11.1 and 3.11.2, the results change drastically. Starting from the lower bound for the qualities, the algorithm converges to the optimal solution in 8 iterations, and solves a total of only 8 subproblems. Thus, applying the additional properties results in an improvement of over 2 orders of magnitude. Similar reductions were obtained for other starting points.

3.12 THE GOP IN A BRANCH AND BOUND FRAMEWORK

In this section, a branch-and-bound framework of the GOP algorithm proposed by Visweswaran and Floudas (1996a) is presented, along with several reduction tests that can be applied at each node of the branch-and-bound tree. The effect

of the properties is to prune the tree and to provide tighter underestimators for the *relaxed dual* problems.

3.12.1 Primal and Relaxed Dual Problems

As was discussed in sections 3.2 and 3.3, the GOP algorithm utilizes primal and relaxed dual subproblems to obtain upper and lower bounds on the global solution. The primal problem results from fixing the $\boldsymbol{y}$ variables to some value, say $\boldsymbol{y}^k$, and takes the form of (3.2). Notice that because of the introduction of additional constraints by fixing the $\boldsymbol{y}$ variables, this problem provides an upper bound on the global optimum of (3.1). Moreover, $P^k(\boldsymbol{y}^k)$, the solution value of this problem yields a solution $\boldsymbol{x}^k$ for the x variables and Lagrange multipliers λ^k and μ^k for the equality and inequality constraints respectively. Note also that it is assumed here that the primal problem is feasible for $\boldsymbol{y} = \boldsymbol{y}^k$.

The Lagrange function constructed from the primal problem is given as:

$$L^k(\boldsymbol{x}, \boldsymbol{y}, \lambda^k, \mu^k) = f(\boldsymbol{x}, \boldsymbol{y}) + {\lambda^k}^T \boldsymbol{h}(\boldsymbol{x}, \boldsymbol{y}) + {\mu^k}^T \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{y}). \tag{3.56}$$

The $\boldsymbol{x}$ variables that are present in the linearization of the Lagrange function around $\boldsymbol{x}^k$, and for which the gradients of the Lagrange functions with respect to $\boldsymbol{x}$ at $\boldsymbol{x}^k$ are functions of the $\boldsymbol{y}$-variables, are called the *connected* variables. The linearization of the Lagrange function around $\boldsymbol{x}^k$ can be written in the form:

$$L^k(\boldsymbol{x}, \boldsymbol{y}, \lambda^k, \mu^k)\big|_{\boldsymbol{x}^k}^{lin} = L_0^k(\boldsymbol{y}, \lambda^k, \mu^k) + \sum_{i=1}^{NI_C^k} x_i g_i^k(\boldsymbol{y}) \tag{3.57}$$

where NI_C^k is the number of *connected* variables at the *kth* iteration (representing the $\boldsymbol{x}$ variables that appear in the Lagrange function), and $L_0^k(\boldsymbol{y}, \lambda^k, \mu^k)$ denotes all the terms in the linearized Lagrange function that depend only on $\boldsymbol{y}$. The positivity and negativity of the functions $g_i^k(\boldsymbol{y})$ define a set of equations that are called the *qualifying* constraints of the Lagrange function at the kth iteration, and which partition the $\boldsymbol{y}$ variable space into $2^{NI_c^k}$ subregions. In each of these subregions, a Lagrange function can be constructed (using the bounds for the $\boldsymbol{x}$ variables) that underestimates the global solution in the subregion, and can therefore be minimized to provide a lower bound for the global solution in that region.

Consider the first iteration of the GOP algorithm. The initial *parent* region is the entire space $\boldsymbol{y} \in Y$ from the original problem. This region is subdivided into $2^{NI_c^1}$ subregions, and in each of these subregions, a subproblem of the following form is solved:

$$\min_{\boldsymbol{y} \in Y, \mu_B} \quad \mu_B$$

$$\begin{aligned} \mu_B \;\geq\; & L^1(\boldsymbol{x}^{B_l}, \boldsymbol{y}, \lambda^1, \mu^1)\big|_{\boldsymbol{x}^1}^{lin}, \\ & \left.\begin{array}{lll} g_i^1(\boldsymbol{y}) \geq 0 & \text{if} & x_i^{B_l} = x_i^L \\ g_i^1(\boldsymbol{y}) \leq 0 & \text{if} & x_i^{B_l} = x_i^U \end{array}\right\} \;\forall\, i \in I_c^1, \end{aligned} \tag{3.58}$$

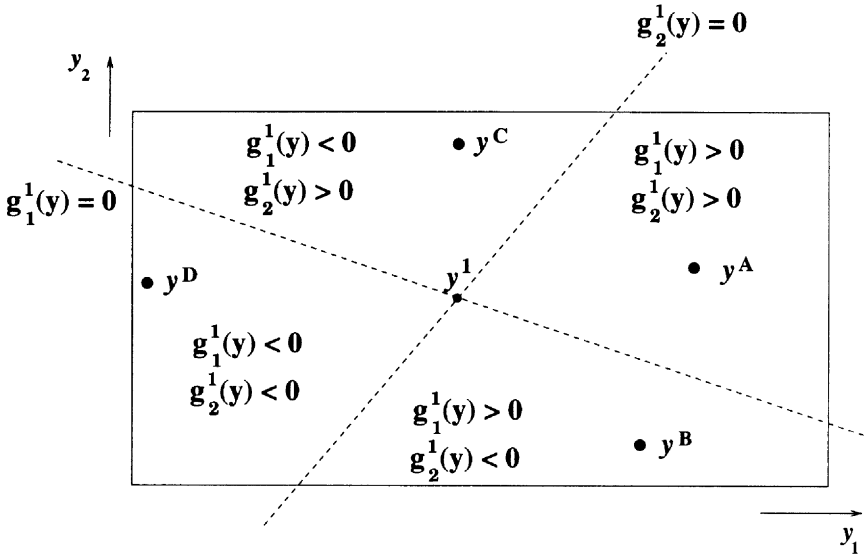


Figure 3.9a. Partition in y for first iteration with two *connected* variables

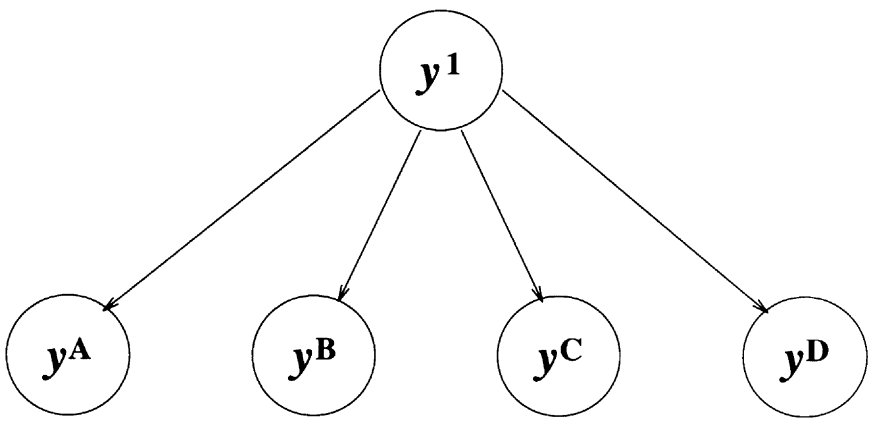


Figure 3.9b. Branch and bound tree for first iteration

where I_c^1 is the set of *connected* variables at the first iteration, NI_C^k is the number of *connected* variables, and x_i^L and x_i^U are the lower and upper bounds on the ith *connected* variable respectively. This subproblem corresponds to the minimization of the Lagrange function, with the *connected* variables replaced

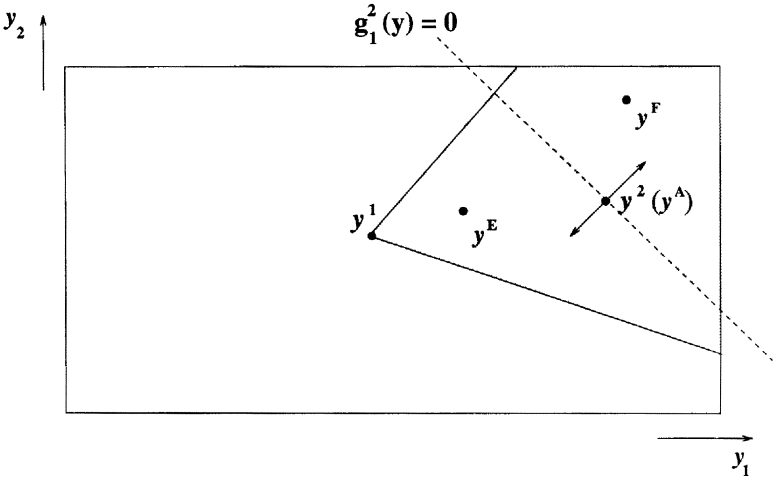


Figure 3.10a. Partition in y for second iteration with one *connected* variable

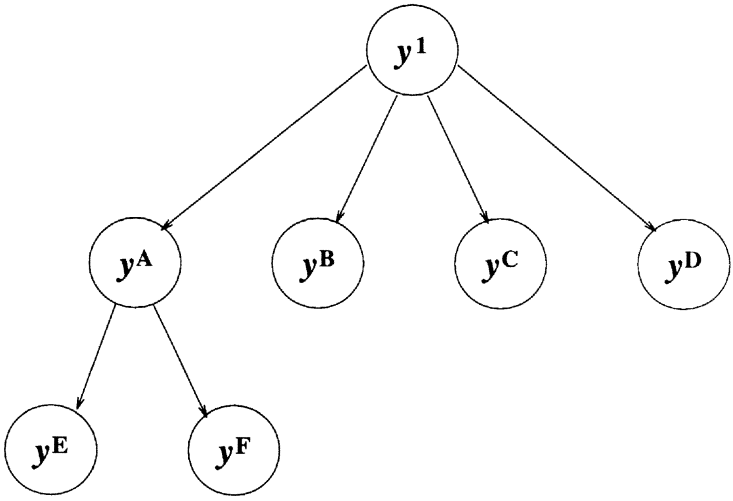


Figure 3.10b. Branch and bound tree for second iteration

by a combination of their lower and upper bounds. Note the presence of the *qualifying* constraints in the problem. These constraints ensure that the minimization is carried out in a subregion of the *parent* node. If this problem has a value of μ_B that is lower than the current best upper bound obtained from

the primal problem, then it is added to the set of candidate lower bounds; otherwise, the region is fathomed, that is, removed from consideration for further refinement.

Consider a problem with two $\boldsymbol{x}$ and two $\boldsymbol{y}$ variables. In the first iteration, assuming that both x_1 and x_2 are in the set of *connected* variables for the first iteration, there are four relaxed dual subproblems solved. These problems are shown in Figure 3.9a. It can be seen that the *qualifying* constraints partition the $\boldsymbol{y}$-space into four regions. Each of the relaxed dual subproblems solved provides a valid underestimator for the corresponding region, as well as a solution point (denoted in the figure by $\boldsymbol{y}^A$, $\boldsymbol{y}^B$, $\boldsymbol{y}^C$ and $\boldsymbol{y}^D$) in the region.

Figure 3.9b shows the corresponding branch-and-bound tree created by the solution of these four problems. The starting point $\boldsymbol{y}^1$ is the root node, and it spawns four leaf nodes. The infimum of the four nodes provides the point for the next iteration, in this case, say $\boldsymbol{y}^A$.

In the second iteration, the relaxed dual problem is equivalent to further partitioning the subregion that was selected for refinement. In each of these partitions, a *relaxed dual* subproblem is solved. Figure 3.10a shows the subregions created in the example, assuming that there was only one *connected* variable in this iteration. The two relaxed dual subproblems solved in this iteration give new solutions $\boldsymbol{y}^E$ and $\boldsymbol{y}^F$ and are possible candidates for entering at future iterations. Figure 3.10b shows the corresponding nodes in the branch-and-bound tree created by this iteration.

The preceding discussion illustrates the key features of a branch and bound framework for the algorithm. The framework is based upon the successive refinement of regions by partitioning on the basis of the *qualifying* constraints. In the next section, the key features of its implementation are discussed, based on which a formal statement of the algorithm is then presented.

3.12.2 A Branch-and-Bound Framework for the GOP Algorithm

The terminology used in this section is as follows. Given a node j in the branch and bound tree, P_j is its parent node, and I_j is the iteration at which node j is created. R_j is the set of constraints defining the region corresponding to node j. At any point, N denotes the total number of nodes in the tree, and C denotes the current node.

3.12.2.1 Root Node and Starting Region. At the beginning of the algorithm, there are no subdivisions in the $\boldsymbol{y}$-space. Therefore, the root node in the branch and bound tree is simply the starting point for the algorithm, $\boldsymbol{y}^1$. The region of application for this node (i.e., the *current* region) is the entire $\boldsymbol{y}$-space.

3.12.3 Reduction Tests at Each node

At each node, the current region of application is divided into several subregions using the *qualifying* constraints of the current Lagrange function. It is possible

to conduct simple tests on the basis of the signs of the *qualifying* constraints that can be used to reduce the number of *connected* variables. One such test, based upon the properties discussed in section 3.11 (see also Visweswaran and Floudas (1993)) is presented below:

Reduction Test:

Suppose a node j is to be partitioned in the kth iteration (i.e., $I_j = k$). Then,

(i) If $g_i^k(\boldsymbol{y}) \geq 0 \quad \forall y \in R_j$, set $x_i = x_i^L$ in $L^k(\boldsymbol{x}, \boldsymbol{y}, \lambda^k, \mu^k)$ and remove i from the set of *connected* variables.

(ii) If $g_i^k(\boldsymbol{y}) \leq 0 \quad \forall y \in R_j$, set $x_i = x_i^U$ in $L^k(\boldsymbol{x}, \boldsymbol{y}, \lambda^k, \mu^k)$ and remove i from the set of *connected* variables.

3.12.4 Evaluation of bounds for the x variables

Often, the original problem contains linear and/or convex constraints in both $\boldsymbol{x}$ and $\boldsymbol{y}$. When the relaxed dual problem is being solved at a given iteration, the region for the $\boldsymbol{y}$ variables is smaller than for the original problem. This can be exploited to provide tighter bounds on the $\boldsymbol{x}$ variables.

Consider, for example, a problem where there is a one-to-one correspondence between the $\boldsymbol{x}$ and $\boldsymbol{y}$ variable set in the feasible region of the problem (i.e., $y_i = x_i$). Then, consider the Kth iteration of the GOP algorithm, where the node j is being partitioned. R_j is the set of constraints defining the current region. Then, it is possible to obtain tighter bounds on the $\boldsymbol{x}$ variables by the following procedure:

1. Choose an $i \in I_c^K$.
2. Solve the following two problems:

$$\begin{aligned} \min_{\boldsymbol{x}} \quad & \pm x_i \\ \boldsymbol{x} - \boldsymbol{y} &= \boldsymbol{0} \\ \boldsymbol{y} &\in R_j. \end{aligned} \tag{3.59}$$

   Use the solutions of the two problems for the lower and upper bounds on x_i respectively. Note that the set R_j includes all linear and convex constraints from the original problem.

3. Repeat Step 1 and 2 for all $i \in I_c^K$.

Similarly, when there are other convex constraints in $\boldsymbol{x}$ and $\boldsymbol{y}$, these constraints can be added to the above problem. This procedure can be very useful in obtaining the tightest bounds on the *connected* $\boldsymbol{x}$ variables at each iteration and consequently in obtaining the tightest underestimators for the relaxed dual subproblems. Note also that in the case of nonconvex constraints, we can incorporate their convex underestimators in the evaluation of the bounds problems.

3.12.4.1 Branch-and-Bound GOP Algorithm. The major steps of the branch-and-bound version of the GOP algorithm are described in this section. The terminology is the same as described in section 3.12.2. In addition, F denotes the set of iterations with a feasible primal problem, while I denotes the set of iterations when the primal problem was infeasible.

STEP 0: Initialization

(a) Read in the data for the problem including tolerance for convergence, ϵ.

(b) Define initial upper and lower bounds (f^U, f^L) on the global optimum.

(c) Generate initial bounds for the x variables, $\boldsymbol{x}^L$ and $\boldsymbol{x}^U$.

(d) Choose a starting point $\boldsymbol{y}^1$ for the algorithm.

(e) Set $K = 1$, $C = P_C = 1$, $N = 1$.

STEP 1: Selection of Current Region

(a) If $K = 1$, set $R_C = \emptyset$ and goto Step 2.

(b) If $K \geq 2$, set $R_C = \emptyset$, $m = C$. Then:

  (i) Add the Lagrange function and *qualifying* constraints for node m to R_C.

  (ii) Set $m = P_m$. If $m = 1$, then goto Step 2.

  (iii) Repeat steps (i) and (ii).

STEP 2: Primal problem

(a) Solve the primal problem (3.2) to give $P^K(\boldsymbol{y}^K)$.

  (i) If feasible, set $F = F \cup K$ and update $f^U = MIN[f^U, P^K(\boldsymbol{y}^K)]$.

  (ii) If infeasible, solve a *relaxed* primal problem. Set $I = I \cup K$.

(b) Store $\boldsymbol{y}^K$, λ^K and μ^K.

STEP 3: Determination of Current Partitions

(a) Generate the current Lagrange function $L^K(\boldsymbol{x}, \boldsymbol{y}, \lambda^K, \mu^K)$.

(b) Determine the set of *connected* variables I_c^K and the corresponding partial derivatives $g_i^K(\boldsymbol{y})$ $(i = 1, \ldots, I_c^K)$ of the current Lagrange function.

(c) For each *connected* variable, determine (if possible) tight lower and upper bounds x_i^L and x_i^U in the current region $\boldsymbol{y} \in R_C$. Otherwise, use the original bounds.

(d) Evaluate lower and upper bounds on $g_i^K(\boldsymbol{y})$ in the region $\boldsymbol{y} \in R_C$.

(i) If $g_i^K(\boldsymbol{y}) \geq 0 \;\; \forall \boldsymbol{y} \in R_C$, set $x_i^B = x_i^L$ in the current Lagrange function, and remove i from the set I_c^K.

(ii) If $g_i^K(\boldsymbol{y}) \leq 0 \;\; \forall \boldsymbol{y} \in R_C$, set $x_i^B = x_i^U$ in the current Lagrange function, and remove i from the set I_c^K.

STEP 4: Relaxed Dual Problem

(a) Select a combination of the bounds B_l of the *connected* variables, say $B_l = B_1$.

(b) Find the solution $(\mu_B^\star, \boldsymbol{y}^\star)$ to the following *relaxed dual* subproblem:

$$\min_{\boldsymbol{y} \in Y, \mu_B} \mu_B$$

$$\begin{aligned} \mu_B &\geq L^K(\boldsymbol{x}^{B_l}, \boldsymbol{y}, \lambda^K, \mu^K)\big|_{\boldsymbol{x}^K}^{lin} \\ & \quad g_i^K(\boldsymbol{y}) \geq 0 \quad if\ x_i^{B_l} = x_i^L \\ & \quad g_i^K(\boldsymbol{y}) \leq 0 \quad if\ x_i^{B_l} = x_i^U \\ (\boldsymbol{y}, \mu_B) &\in R_C \end{aligned} \tag{3.60}$$

(i) If $\mu_B^\star < f^U - \epsilon$, set $j = N+1, P(j) = C, N = N+1$, and store the solution in μ_B^j, y^j.

(ii) If $\mu_B^\star \geq f^U - \epsilon$, fathom the solution.

(c) Select a new combination of bounds, say $B_l = B_2$, for the *connected* variables.

(d) Repeat steps (b) and (c) until all the combinations of bounds for the *connected* variables have been considered.

STEP 5: Selecting a new lower bound
Select the infimum of all μ_B^j, say μ_B^p. Set $C = p, \boldsymbol{y}^{K+1} = \boldsymbol{y}^p$, $f^L = \mu_B^p$.

STEP 6: Check for convergence
If $\left|\frac{f^U - f^L}{f^U}\right| < \epsilon$, STOP; otherwise, set $K = K+1$ and return to step 1.

3.12.4.2 Illustration. Consider the application of the branch and bound algorithm to the following problem, taken from Al-Khayyal and Falk (1983):

$$\begin{aligned} \min_{x,y} \; & -x + xy - y \\ -6x + 8y - 3 &\leq 0 \\ 3x - y - 3 &\leq 0 \\ x, y &\geq 0. \end{aligned}$$

Note that with these constraints, the bounds on both x and y are $[0, 1.5]$. Consider a starting point of $y^1 = 1$ for the algorithm.

Iteration 1 For $y^1 = 1$, the first primal problem has the solution of $x = 0$, $\mu_1^1 = \mu_2^1 = \mu_3^1 = 0$, with the objective value of -1. The upper bound on the problem is therefore -1. The Lagrange function is given by

$$L^1(x, y, \mu^1) = -x + xy - y = xg_1^1(y) - y$$

where $g_1^1(y) = y - 1$ is the first (and only) *qualifying* constraint. From the original problem, the bounds on y are $[0, 1.5]$. Therefore, $-1.0 \leq g_1^1(y) \leq 0.5$, implying that two relaxed dual subproblems need to be solved. These problems, solved for positive and negative $g_1^1(y)$ (with x set to 0 and 1.5 respectively), are shown below:

<u>Node 1</u>

$$\begin{aligned} \min_{y,\mu_B} \quad & \mu_B \\ \mu_B &\geq -y \\ g_1^1(y) &= y - 1 \geq 0 \\ 0 &\leq y \leq 1.5 \end{aligned}$$

Solution: $y = 1.5$, $\mu_B = -1.5$

<u>Node 2</u>

$$\begin{aligned} \min_{y,\mu_B} \quad & \mu_B \\ \mu_B &\geq 0.5y - 1.5 \\ g_1^1(y) &= y - 1 \leq 0 \\ 0 &\leq y \leq 1.5 \end{aligned}$$

Solution: $y = 0.0$, $\mu_B = -1.5$

The solution of these two problems provides the first partition in the tree depicted in Figure 3.11. R is the root node corresponding to the starting point $y^1 = 1.0$. At this iteration, two nodes 1 and 2 are created by the solution of the two relaxed dual subproblems. Both nodes have the root node R as their *parent* node. Both problems have equal objective values, and are thus equal candidates for the best lower bound. Suppose that node 2 is selected for further exploration.

Iteration 2 From node 2, the current value of y is 0.0. For this value, the primal problem has the solution $x = 1.0$, $\mu_1^2 = 0$, $\mu_2^2 = \frac{1}{3}$, $\mu_3^2 = 0$, with an objective value of -1.0. The Lagrange function from this problem is

$$L^2(x, y, \mu^2) = -x + xy - y + \frac{1}{3}(3x - y - 3) = xg_1^2(y) - \frac{4}{3}y - 1$$

where $g_1^2(y) = y$ is the *qualifying* constraint for this Lagrange function.

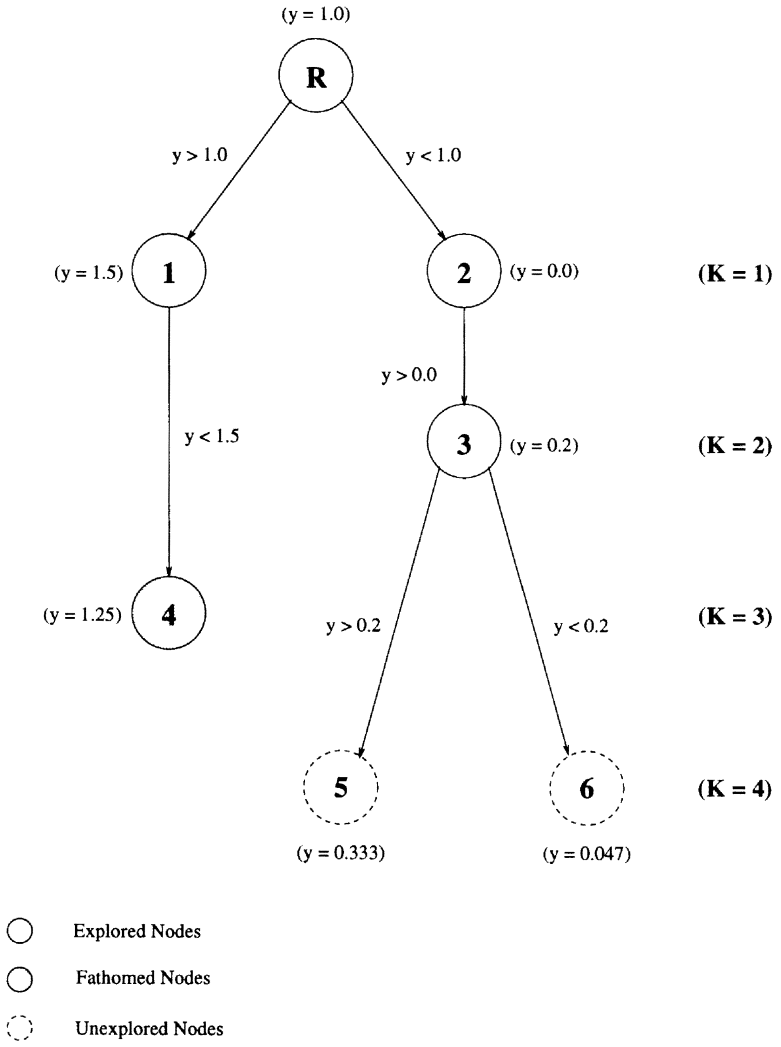


Figure 3.11. Branch and Bound Tree for Illustrative Example

For this iteration, the relaxed dual subproblems are solved in the region $0 \leq y \leq 1$. The tightest bounds on x for this region can be found by solving

the following two problems:

$$\begin{aligned} \min_x \quad & \pm x \\ -6x + 8y - 3 &\leq 0 \\ 3x - y - 3 &\leq 0 \\ 0 \leq y &\leq 1 \\ x &\geq 0. \end{aligned}$$

The solutions of these problems provide lower and upper bounds for x respectively. Thus, for the region $0 \leq y \leq 1$, this yields the bounds $0 \leq x \leq \frac{4}{3}$.

Since $y > 0$, it is obvious that $g_1^2(y)$ is positive for all y in the current region. Therefore, only one relaxed dual problem needs to be solved, with a valid underestimator to $L^2(x, y, \mu^2)$ being used by fixing x to its lower bound. Moreover, from the first iteration, the Lagrange function corresponding to node 2 is also a valid cut for this region. Note, however, that instead of using the original bounds on x in both these Lagrange functions, the improved bounds can be used. This yields the following relaxed dual problem:

<u>Node 3</u>

$$\begin{aligned} \min_{y,\mu_B} \quad & \mu_B \\ \mu_B &\geq \tfrac{1}{3}y - \tfrac{4}{3} \\ g_1^1(y) &= y - 1 \leq 0 \\ \mu_B &\geq -\tfrac{4}{3}y - 1 \\ 0 &\leq y \leq 1.5 \end{aligned}$$

Solution: $y = 0.2$, $\mu_B = -1.26667$.

At the end of this iteration, there are two candidate regions for further partitioning: (i) the region $1 \leq y \leq 1.5$ corresponding to node 1, with a lower bound of -1.5, and (ii) the region $0 \leq y \leq 1$ corresponding to node 3, with the lower bound of -1.26667. Following the criterion of selecting the region with the best lower bound, node 1 is chosen for further exploration.

Iteration 3 From node 1, the current value of y is 1.5. For this value, the primal problem has the solution $x = 1.5$, $\mu_1^3 = \frac{1}{12}$, $\mu_2^3 = 0$, $\mu_3^3 = 0$, with the objective value of -0.75. The Lagrange function for this iteration is

$$L^3(x, y, \mu^3) = -x + xy - y + \frac{1}{12}(-6x + 8y - 3) = xg_1^3(y) - \frac{1}{3}y - \frac{1}{4}$$

where $g_1^3(y) = y - 1.5$ is the *qualifying* constraint for this Lagrange function.

For this iteration, the relaxed dual subproblems are solved in the region $1 \leq y \leq 1.5$, In this region, solving the bounds problems for x yields $\frac{5}{6} \leq x \leq 1.5$. Since $g_1^3(y) \leq 0$ for all y, only one relaxed dual problem needs to be solved, with x fixed to its upper bound. From the first iteration, the Lagrange function

corresponding to node 1 is also a valid cut for this region. Using the improved bounds on x shown above yields the following relaxed dual problem:

Node 4

$$\begin{array}{rcl} \min\limits_{y,\mu_B} & & \mu_B \\ \mu_B & \geq & -\frac{1}{6}y - \frac{5}{6} \\ g_1^1(y) & = & y - 1 \geq 0 \\ \mu_B & \geq & \frac{7}{6}y - 2.5 \\ 0 & \leq & y \leq 1.5 \end{array}$$

Solution: $y = 1.25$, $\mu_B = -1.04167$.

Again, there is no partition of the region in this iteration, but the relaxed dual provides a tighter lower bound for this region than was originally available.

At the end of this iteration, there are two candidate regions for further partitioning: (i) the region $0 \leq y \leq 1$, corresponding to node 3, with the lower bound of -1.26667, and (ii) the region $1 \leq y \leq 1.5$, corresponding to node 4, with the lower bound of -1.04167. Following the criterion of selecting the region with the best lower bound, node 3 is chosen for further exploration.

Iteration 4 From node 3, the current value of y is 0.2. For this value, the primal problem has the solution $x = 1.0667$, $\mu_1^4 = 0$, $\mu_2^4 = 0.2667$, $\mu_3^4 = 0$, with the objective value of -1.05333. Note that the solution of this problem has the immediate consequence that it provides an upper bound that is lower than the lower bound for node 4 (which is -1.04167). Therefore, node 4 can be immediately fathomed, that is, removed from consideration for any further refinement or exploration.

The Lagrange function from the current primal problem is

$$L^4(x, y, \mu^4) = -x + xy - y + 0.2667(3x - y - 3) = xg_1^4(y) - 1.2667y - 0.8$$

where $g_1^4(y) = y - 0.2$ is the *qualifying* constraint for this Lagrange function.

For this iteration, the relaxed dual subproblems are solved in the region $0 \leq y \leq 1.0$, and try to provide refined lower bounds by partitioning the region further. The tightest bounds for x in this region are $0 \leq x \leq \frac{4}{3}$.

Unlike the previous two iterations, it is necessary to partition the current region since $-0.2 \leq g_1^4(y) \leq 1.3$ and the reduction tests of Section 3.12.3 do not provide any help. It is therefore necessary to solve two relaxed dual subproblems in the current iteration. For both these problems, the Lagrange functions from nodes 2 and 3 are valid underestimators. These two problems are shown below:

Node 5

$$\begin{aligned} \min_{y,\mu_B} \quad & \mu_B \\ \mu_B &\geq \tfrac{1}{3}y - \tfrac{4}{3} \\ g_1^1(y) &= y - 1 \leq 0 \\ \mu_B &\geq -\tfrac{4}{3}y - 1 \\ \mu_B &\geq -1.2667y - 0.8 \\ g_1^4(y) &= y - 0.2 \geq 0 \\ 0 &\leq y \leq 1.5 \end{aligned}$$

Solution: $y = 0.333$, $\mu_B = -1.2222$.

Node 6

$$\begin{aligned} \min_{y,\mu_B} \quad & \mu_B \\ \mu_B &\geq \tfrac{1}{3}y - \tfrac{4}{3} \\ g_1^1(y) &= y - 1 \leq 0 \\ \mu_B &\geq -\tfrac{4}{3}y - 1 \\ \mu_B &\geq 0.0667y - 1.0667 \\ g_1^4(y) &= y - 0.2 \leq 0 \\ 0 &\leq y \leq 1.5 \end{aligned}$$

Solution: $y = 0.04762$, $\mu_B = -1.06349$.

Together, these two problems provide a tighter lower bound (-1.2222) for the region $0 \leq y \leq 1$ than before (-1.26667).

At the end of this iteration, there are two candidate regions for further partitioning – the region $0 \leq y \leq 0.2$, corresponding to node 6, with the lower bound of -1.06349, and (ii) the region $0.2 \leq y \leq 1$, corresponding to node 5, with the lower bound of -1.2222. Therefore, node 5 is chosen for further refinement.

The algorithm continues in this fashion for 18 iterations, converging to the global solution of -1.0833 at $x = 1.1667, y = 0.5$ with a tolerance of 0.001 between the upper and lower bounds. It is interesting to note that the original GOP algorithm, which does not compute the tightest bounds on the x variables at each iteration, takes 76 iterations to converge with the same tolerance. This indicates the importance of having the tightest possible bounds on the *connected* variables at each iteration.

3.13 REFORMULATION OF THE RELAXED DUAL AS A SINGLE MILP PROBLEM

The solution of the relaxed dual subproblems at each node is the most time-consuming step in the algorithm outlined in Section 3.12.2. The reduction test mentioned in Section 3.12.3 can help to prune the branch-and-bound tree at each node; however, it is still necessary to solve a large number of subproblems at each iteration. It is very likely that the solution of most of these subproblems is not needed as far as the succeeding iterations are concerned, that is, most of the nodes will be fathomed as soon as they are spawned. Naturally, this raises the question whether these subproblems can be solved implicitly. This section presents one possible approach for reformulation of the relaxed dual problem at each iteration so that the implicit enumeration of all the solutions can be achieved by solution of an MILP problem.

At the Kth iteration, the Lagrange function has the form given by (3.6). Consider the ith term in the summation. In each of the $2^{NI_C^k}$ relaxed dual subproblems, this term takes on either of two values:

$$x_i g_i^K(\boldsymbol{y}) = \begin{cases} x_i^L g_i^K(\boldsymbol{y}) & \text{if} \quad g_i^K(\boldsymbol{y}) \geq 0 \\ x_i^U g_i^K(\boldsymbol{y}) & \text{if} \quad g_i^K(\boldsymbol{y}) \leq 0. \end{cases}$$

Now, x_i can be implicitly expressed as a combination of its lower and upper bounds:

$$x_i = (1 - \alpha_i^K) x_i^L + \alpha_i^K x_i^U \tag{3.61}$$

where $\alpha_i^K \in \{0, 1\}$.

This leads to the following formulation for the ith term in (3.6):

$$x_i g_i^K(\boldsymbol{y}) = t_i + x_i^L g_i^K(\boldsymbol{y}) \tag{3.62}$$

where

$$t_i \geq \alpha_i^K (x_i^U - x_i^L) \underline{g_i^K} \tag{3.63}$$

$$t_i \geq (x_i^U - x_i^L)(g_i^K(y) - (1 - \alpha_i^K)\overline{g_i^K}) \tag{3.64}$$

$$\alpha_i^K \underline{g_i^K} \leq g_i^K(\boldsymbol{y}) \leq (1 - \alpha_i^K)\overline{g_i^K} \tag{3.65}$$

where $\underline{g_i^K}$ and $\overline{g_i^K}$ are respectively the lower and upper bounds on the *qualifying* constraints. As the following property shows, this can be used to reformulate the relaxed dual problem as a mixed integer linear program (MILP):

Property 3.13.1 *Suppose that, at the Kth iteration, C denotes the current node to be partitioned, and R_C denotes the set of constraints defining the region associated with C. Then, the best solution from all the relaxed dual subproblems at this iteration can be obtained as the optimal solution of the following mixed-integer linear program.*

$$\min_{\substack{\boldsymbol{y} \in Y, \mu_B \\ t, \alpha}} \mu_B \tag{3.66}$$

$$\mu_B \geq \sum_{i=1}^{NI_c^K} t_i^K + \sum_{i=1}^{NI_c^K} x_i^L g_i^K(\boldsymbol{y}) + L_0^K(\boldsymbol{y}, \lambda^K, \mu^K) \tag{3.67}$$

$$t_i^K \geq \alpha_i^K (x_i^U - x_i^L) \underline{g_i^K} \tag{3.68}$$

$$t_i^K \geq (x_i^U - x_i^L)(g_i^K(\boldsymbol{y}) - (1 - \alpha_i^K)\overline{g_i^K}) \tag{3.69}$$

$$\alpha_i^K \underline{g_i^K} \leq g_i^K(\boldsymbol{y}) \leq (1 - \alpha_i^K)\overline{g_i^K} \tag{3.70}$$

$$t^K \in \Re^{NI_c^K}, \ \alpha^K \in \{0, 1\}^{NI_c^K}, \ \boldsymbol{y} \in Y \tag{3.71}$$

$$(\boldsymbol{y}, \mu_B) \in R_C \tag{3.72}$$

where $\underline{g_i^K}$ and $\overline{g_i^K}$ are the lower and upper bounds on $g_i^K(\boldsymbol{y})$ over Y.

Proof. Since α_i^K is a binary variable, it can take on only two values in any solution, either 0 or 1. Consider these two possible cases for α_i^K:

Case I ($\alpha_i^K = 0$) **:**
In this case, equations (3.68)-(3.70) reduce to

$$t_i^K \geq 0 \tag{3.73}$$
$$t_i^K \geq (x_i^U - x_i^L)(g_i^K(\boldsymbol{y}) - \overline{g_i^K}) \tag{3.74}$$
$$0 \leq g_i^K(\boldsymbol{y}) \leq \overline{g_i^K}. \tag{3.75}$$

Since $g_i^K(\boldsymbol{y}) \leq \overline{g_i^K}$ for all $\boldsymbol{y} \in Y$, (3.74) is redundant. Similarly, the second inequality in (3.75) is also trivially satisfied. Therefore, if this set of constraints is active in any solution, then $t_i^K = 0$, the contribution from the ith components of the first two terms in (3.67) to μ_B is $x_i^L g_i^K(\boldsymbol{y})$, and in addition, we must also have $g_i^K(\boldsymbol{y}) \geq 0$.

Case II ($\alpha_i^K = 1$) **:**
In this case, equations (3.68)-(3.70) reduce to

$$t_i^K \geq (x_i^U - x_i^L)\underline{g_i^K} \tag{3.76}$$
$$t_i^K \geq (x_i^U - x_i^L)g_i^K(\boldsymbol{y}) \tag{3.77}$$
$$\underline{g_i^K} \leq g_i^K(\boldsymbol{y}) \leq 0. \tag{3.78}$$

Since $g_i^K(\boldsymbol{y}) \geq \underline{g_i^K}$ for all $\boldsymbol{y} \in Y$, (3.76) is redundant. Similarly, the first inequality in (3.78) is trivially satisfied. Therefore, if this set of constraints is active in any solution, then $t_i^K = (x_i^U - x_i^L)g_i^K(\boldsymbol{y})$, the contribution from the ith components of the first two terms in (3.67) to μ_B is $x_i^U g_i^K(\boldsymbol{y})$, and in addition, $g_i^K(\boldsymbol{y}) \leq 0$.

Thus, it can be seen that any solution of the relaxed dual problem in Step 4 of the algorithm in Section 3.12.2 is automatically embedded in the set of constraints described by (3.67)-(3.72). Therefore, (3.66)-(3.72) provide a valid formulation for obtaining the solution of the relaxed dual problem.

Remark 3.13.1 If $L_0^K(\boldsymbol{y}, \lambda^K, \mu^K)$ are convex functions in $\boldsymbol{y}$, then (3.66)-(3.72) is a convex MINLP, and can be solved with the Generalized Benders Decomposition (Geoffrion, 1972; Floudas *et al.*, 1989) or the Outer Approximation algorithm (Duran and Grossmann, 1986a). For a detailed discussion of the MINLP approaches the reader is directed to the textbook of Floudas (1995).

It should be noted that the reduction tests of Section 3.12.3 can also be applied to the MILP formulation, as shown by the following property.

Property 3.13.2 *At the Kth iteration,*

(i) *If $g_i^K(\boldsymbol{y}) \geq 0$ for all $\boldsymbol{y}$ (respectively $g_i^K(\boldsymbol{y}) \leq 0$ for all $\boldsymbol{y}$) then variable α_i^K can be fixed to 0 (respectively 1.)*

(ii) *If $g_i^K(y) = 0$ for all y then variable α_i^K vanishes from formulation (3.66)-(3.72).*

Proof. **(i)** Suppose that $g_i^K(y) \geq 0$ for all $y \in Y$. Then, to underestimate the Lagrange function from the Kth iteration, x_i^B must be set to x_i^L. By the definition of α_i^K,

$$x_i = (1 - \alpha_i^K)x_i^L + \alpha_i^K x_i^U.$$

Hence, this leads to $\alpha_i^K = 0$. Conversely, if $g_i^K(y) \leq 0$ for all $y \in Y$, then α_i^K must be equal to 1.
(ii) If $g_i^K(y) = 0$ for all $y \in Y$, then this implies that

$$\underline{g_i^K} = g_i^K(y) = \overline{g_i^K} = 0.$$

Therefore, in (3.66)-(3.72), t_i^K is always equal to zero, and the variable α_i^K vanishes from the formulation.

Backtracking With the MILP reformulation, it is possible to solve the relaxed dual subproblems implicitly for the best solution at each iteration. However, it is not sufficient to find the best solution; it must also be determined whether any of the other partitions can provide a useful solution for further refinement.

Consider the relaxed dual subproblems solved when node j is being partitioned. Suppose that this node was partitioned during iteration K. Then, there are NI_c^K binary variables, and $2^{NI_c^K}$ partitions to consider. Solving the problem (3.66)-(3.72) gives the best solution among these partitions. Suppose that this solution corresponds to the combination α^C. Suppose also that J_C is the set of binary variables that are equal to 1 in this combination, and that there are NJ_C of them. Consider now the following cut

$$\sum_{i \in J_C} \alpha_i - \sum_{i \notin J_C} \alpha_i \leq NJ_C - 1.$$

If problem (3.66)-(3.72) is resolved with the above cut added to the problem, then the solution will have a value for α different from α^C, and will therefore correspond to a different subregion of the current problem. Note that the objective value of this problem represents the "second" best possible solution. The best solution, of course, is the one corresponding to the solution of the first MILP problem, with $\alpha = \alpha^C$. Therefore, this methodology is sufficient to go back to a partitioned node at any point.

Note that although the size of the MILP problems increases slightly at each iteration due to the accumulation of constraints from previous iterations, the number of binary variables present in these problems is equal to the number of *connected* variables for each iteration. In other words, the number of binary variables in the MILP problems is bounded by the number of x variables in the original problem.

3.13.1 *The* GOP/MILP *Algorithm*

The terminology is as described in the previous section. In addition, A_j denotes the set of integer cuts to be used when solving the MILP problem for the node j.

STEP 0: Initialization
This step is the same as in Section 3.12.4.1, with the addition of setting $A_1 = \emptyset$.

STEP 1 – Step 3:
Same as in Section 3.12.4.1.

STEP 4: Current Relaxed Dual Problem
Solve the MILP problem (3.66)-(3.72).

(i) If $\mu_B^\star < f^U - \epsilon$, set $j = N+1, P(j) = C, N = N+1$, and store the solution in μ_B^j, y^j.

(ii) If $\mu_B^\star \geq f^U - \epsilon$, fathom the solution.

Let the solution for the binary variables in this problem be $\alpha = \alpha^C$. Let J_C be the set of variables which are 1 in this solution, and let NJ_C the number of such binary variables.

STEP 5: Select a new lower bound
Same as in Section 3.12.4.1.

STEP 6: Regenerate Solutions From Partitioned Nodes
Suppose that the solution selected in Step 5 corresponds to node C, and that this node was originally partitioned at iteration k. Then, add the cut

$$\sum_{i \in J_C} \alpha_i - \sum_{i \notin J_C} \alpha_i \leq NJ_C - 1$$

to the set of binary cuts A_C. Solve the MILP problem (3.66)-(3.72) with the added set of binary cuts A_C. Suppose the solution of this problem is $\mu_B^\star$.

(i) If $\mu_B^\star < f^U - \epsilon$, then set $j = N+1, P(j) = C, N = N+1$, and store the solution in μ_B^j, y^j. Also set α^C to be the solution of the binary variables in this formulation.

(ii) If $\mu_B^\star \geq f^U - \epsilon$, fathom the node C.

STEP 7: Check for convergence
Same as in Section 3.12.4.1.

Remark 3.13.2 After the MILP problem has been solved in either Step 4 or Step 6, an integer cut is added to the corresponding formulation which ensures that that solution cannot be repeated. This implies that the same MILP formulation might be solved several times over the course of the iterations with small differences arising from the additional integer cuts. Subsequently, there is considerable potential for storing the tree information from these problems for use in future iterations.

Remark 3.13.3 At each iteration of the algorithm, there is a *single* MILP problem solved in Step 4 or Step 6 as compared to the original algorithm, which needs to solve $2^{NI_c^K}$ subproblems at the Kth iteration. This MILP problem contains NI_c^K binary variables in the case of Step 4, or NI_c^L variables in Step 6. In either case, the number of binary variables present in any MILP formulation during all the iterations is bounded by the maximum number of $\boldsymbol{x}$ variables. However, it is usually the case that the number of *connected* variables is a fraction of the total number of $\boldsymbol{x}$ variables, implying that the MILP problems are likely to have few binary variables.

Remark 3.13.4 The major advantage of the MILP problem appears when there are more than about 15 *connected* variables at any iteration. In such cases, the original algorithm would need to solve over 2 million problems at that iteration, the vast majority of which would never be considered as candidate solutions for further branching. In the case of the MILP algorithm, the implicit enumeration allows for far fewer problems to be solved. The maximum number of MILP problems solved is twice the number of iterations of the algorithm.

3.13.2 Illustration of the GOP/MILP *Algorithm*

Consider the example from Section 3.12.4.2, with a starting point of $y^1 = 1$ for the algorithm.

Iteration 1

For $y^1 = 1$, the first primal problem has the solution of $x = 0$, $\mu_1^1 = \mu_2^1 = \mu_3^1 = 0$, with the objective value of -1. The upper bound on the problem is therefore -1. The Lagrange function is given by

$$L^1(x, y, \mu^1) = -x + xy - y = xg_1^1(y) - y$$

where $g_1^1(y) = y - 1$ is the first (and only) *qualifying* constraint.

The following MILP problem is solved first in Step 4:

$$\begin{aligned}
\min_{y,\mu_B} \quad & \mu_B \\
\mu_B &\geq t_1^1 - y \\
t_1^1 &\geq -1.5\alpha_1^1 \\
t_1^1 &\geq 1.5(g_1^1 - 0.5(1-\alpha_1^1)) \\
\alpha_1^1 &\leq g_1^1 \leq 0.5(1-\alpha_1^1) \\
g_1^1 &= y - 1 \\
0 &\leq y \leq 1.5.
\end{aligned}$$

The solution of this problem is $y = 0.0$, $\mu_B = -1.5$, $\alpha_1^1 = 1$. Note that this corresponds to node 2 in the branch and bound tree in Figure 3.11. This solution is chosen to be the next candidate for branching.

Iteration 2

For $y = 0.0$, the primal problem has the solution $x = 1.0$, $\mu_1^2 = 0$, $\mu_2^2 = \frac{1}{3}$, $\mu_3^2 = 0$, with the objective value of -1.0. The Lagrange function from this problem is

$$L^2(x, y, \mu^2) = -x + xy - y + \frac{1}{3}(3x - y - 3) = xg_1^2(y) - \frac{4}{3}y - 1$$

where $g_1^2(y) = y$ is the *qualifying* constraint for this Lagrange function.

Since $0 \leq y \leq 1$, tight bounds on x can be obtained to be $0 \leq x \leq \frac{4}{3}$. Since $y > 0$, a valid underestimator to $L^2(x, y, \mu^2)$ for all y can be obtained by fixing x to its lower bound. Therefore, there are no binary variables, and consequently, the MILP formulation reduces to the same formulation as in Section 3.12.4.1. The solution of the resulting subproblem is $y = 0.2$, $\mu_B = -1.2667$.

At the end of this iteration, there are two candidate regions for further branching: (i) node 1 ($1 \leq y \leq 1.5$) with a lower bound of -1.5, and (ii) node 3 ($0 \leq y \leq 1$) with a lower bound of -1.2667. The former node is selected for further exploration.

Iteration 3

For $y = 1.5$, the primal problem has the solution $x = 1.5$, $\mu_1^3 = \frac{1}{12}$, $\mu_2^3 = 0$, $\mu_3^3 = 0$, with the objective value of -0.75. The Lagrange function from this problem is

$$L^3(x, y, \mu^3) = -x + xy - y + \frac{1}{12}(-6x + 8y - 3) = xg_1^3(y) - \frac{1}{3}y - \frac{1}{4}$$

where $g_1^3(y) = y - 1.5$ is the *qualifying* constraint for this Lagrange function.

For $y \leq 1.5$, the tightest bounds on x are $\frac{5}{6} \leq x \leq 1.5$. Again, only one relaxed dual problem needs to be solved, with a valid underestimator to

$L^3(x, y, \mu^3)$ being used by fixing x to its upper bound. Therefore, the MILP is again identical to the original algorithm formulation, and has the solution $y = 1.25, \mu_B = -1.04167$.

At the end of this iteration, there are two candidate regions for further partitioning - (i) the region $0 \leq y \leq 1$, corresponding to node 3, with a lower bound of -1.26667, and (ii) the region $1 \leq y \leq 1.5$, corresponding to node 4, with the lower bound of -1.04167. Following the criterion of selecting the region with the best lower bound, node 3 is chosen for further exploration.

Iteration 4

For $y = 0.2$, the primal problem has the solution $x = 1.0667$, $\mu_1^4 = 0$, $\mu_2^4 = 0.2667$, $\mu_3^4 = 0$, with the objective value of -1.05333. Note that the solution of this problem provides an upper bound that is lower than the lower bound for node 4 (which is -1.04167). Therefore, node 4 can be immediately fathomed, that is, removed from consideration for any further refinement or exploration.

The Lagrange function from the current primal problem is

$$L^4(x, y, \mu^4) = -x + xy - y + 0.2667(3x - y - 3) = xg_1^4(y) - 1.2667y - 0.8$$

where $g_1^4(y) = y - 0.2$ is the *qualifying* constraint for this Lagrange function.

For this iteration, the relaxed dual subproblems are solved in the region $0 \leq y \leq 1.0$, and try to provide refined lower bounds by partitioning the region further. The tightest bounds for x in this region are $0 \leq x \leq \frac{4}{3}$.

Unlike the previous two iterations, it is necessary to partition the current region since $-0.2 \leq g_1^4(y) \leq 1.3$. Therefore, the MILP in this iteration takes the form:

$$\begin{aligned}
\min_{y,\mu_B} \quad & \mu_B \\
\mu_B &\geq \frac{1}{3}y - \frac{4}{3} \\
g_1^1(y) &= y - 1 \leq 0 \\
\mu_B &\geq -\frac{4}{3}y - 1 \\
\mu_B &\geq t_1^4 - 1.2667y \\
t_1^4 &\geq -0.26667\alpha_1^4 \\
t_1^4 &\geq 1.3333(g_1^4 - (1 - \alpha_1^4) \cdot 1.0667) \\
-0.2\alpha_1^4 &\leq g_1^4 \leq (1 - \alpha_1^4) \cdot 0.8 \\
g_1^4 &= y - 0.2 \\
0 &\leq y \leq 1.5.
\end{aligned}$$

The solution of this problem is $y = 0.333$, $\mu_B = -1.222$, $\alpha_1^4 = 0$.

Thus, the MILP algorithm produces the *exact* sequence of solutions given by the original branch and bound algorithm. As in Section 3.12.4.2, this algorithm also takes 18 iterations to converge.

Remark 3.13.5 Note that in this example, there is no advantage to using the MILP formulation, since it needs to be solved for both combinations of α_1 at each iteration. However, for problems with more than one *connected* variable, it is obvious that this formulation can offer a major advantage over the original formulation. This is because at each iteration, no more than 2 MILP problems need to be solved. Although these problems are larger in size, their structure is such that finding their solution does not depend on the presence of the binary variables, and a good MILP solver can solve them very efficiently. At the same time, they feature the key advantage of not having to solve the full set of subproblems at each iteration.

It should be noted, however, that the convenience of solving just one compact problem is achieved at the expense of problem size. Because all possible solutions of the relaxed dual problem have to be incorporated in the GOP/MILP formulation, the result is a much larger problem to solve. A number of constraints and variables need to be used to implicitly represent all the possible bound combinations.

3.14 A LINEAR BRANCHING SCHEME FOR THE GOP ALGORITHM

In both the GOP and GOP/MILP algorithms, the *qualifying* constraints (i.e., the gradients of the Lagrange function) are used to partition the $\boldsymbol{y}$-space. The reduction properties presented in Section 3.12.2 can provide a significant reduction in the number of *connected* variables and subsequently the number of partitions. However, in the worst case, the number of subproblems solved still increases exponentially with the number of *connected* variables. It is then natural to ask the following question: *Is it possible to develop a valid lower bound at each iteration using only a linearly increasing number of* relaxed dual *subproblems?* In this section, we present one branching scheme that achieves this goal. This scheme originates from the study of Barmish *et al.* (1995) on the stability of polytopes of matrices of robust control systems.

3.14.1 Reformulation of Qualifying Constraints

Consider the relaxed dual problem at the kth iteration. This problem has the constraint

$$\mu_B \geq L_0^k(\boldsymbol{y}, \lambda^k, \mu^k) + \sum_{i=1}^{NI_C^k} x_i g_i^k(\boldsymbol{y}).$$

Suppose that all the $\boldsymbol{x}$ variables are bounded between -1 and 1. If this is not the case, it can be achieved by use of the following linear transformation. Suppose that $\boldsymbol{x}^L \leq \boldsymbol{x} \leq \boldsymbol{x}^U$. Then, define $\boldsymbol{x}'$ such that $-\mathbf{1} \leq \boldsymbol{x}' \leq \mathbf{1}$, and

$$x_i = a_i \cdot x_i' + b_i.$$

The substitution of the lower and upper bounds gives

$$x_i^L = a_i \cdot (-1) + b_i \ , \quad \text{and} \quad x_i^U = a_i \cdot (1) + b_i$$

leading to

$$a_i = \frac{x_i^U - x_i^L}{2}, \quad \text{and} \quad b_i = \frac{x_i^U + x_i^L}{2}.$$

The variables $\boldsymbol{x}'$ can then be substituted for $\boldsymbol{x}$ using the above transformation, leading to a Lagrange function in $\boldsymbol{y}$ and $\boldsymbol{x}'$. We will continue the presentation in this section by considering the case $-\mathbf{1} \leq \boldsymbol{x} \leq \mathbf{1}$.

The following observation is now made:

(a) If $g_i^k(\boldsymbol{y}) \geq 0$,

$$x_i g_i^k(\boldsymbol{y}) \geq x_i^L g_i^k(\boldsymbol{y}) \quad \Rightarrow \quad x_i g_i^k(\boldsymbol{y}) \geq -g_i^k(\boldsymbol{y}).$$

(b) If $g_i^k(\boldsymbol{y}) \leq 0$,

$$x_i g_i^k(\boldsymbol{y}) \geq x_i^U g_i^k(\boldsymbol{y}) \quad \Rightarrow \quad x_i g_i^k(\boldsymbol{y}) \geq g_i^k(\boldsymbol{y}).$$

Combining these two cases leads to the inequality

$$x_i g_i^k(\boldsymbol{y}) \geq -|g_i^k(\boldsymbol{y})|$$

and

$$\mu_B \geq L_0^k(\boldsymbol{y}, \lambda^k, \mu^k) - \sum_{i=1}^{NI_C^k} |g_i^k(\boldsymbol{y})|. \tag{3.79}$$

The first term on the right hand side is convex, and can remain unaltered. Consider now the summation term. Using the concept of the infinity norm, (3.79) can be written as

$$\mu_B \geq L_0^k(\boldsymbol{y}, \lambda^k, \mu^k) - NI_C^k \cdot (\max_i |g_i^k(\boldsymbol{y})|). \tag{3.80}$$

For any value of $\boldsymbol{y}$, there is some $j \in 1, \ldots, NI_C^k$ such that

$$|g_j^k(\boldsymbol{y})| = \max_{i=1,\ldots,NI_C^k} |g_i^k(\boldsymbol{y})|$$

implying that

$$|g_j^k(\boldsymbol{y})| \geq |g_i^k(\boldsymbol{y})|, \quad i = 1, \ldots, NI_C^k. \tag{3.81}$$

Consider the following two possibilities:

(a) If $g_j^k(\boldsymbol{y}) \geq 0$, then $|g_j^k(\boldsymbol{y})| = g_j^k(\boldsymbol{y})$, and (3.81) reduces to the two inequalities

$$\left.\begin{array}{lcl} g_j^k(\boldsymbol{y}) & \geq & g_i^k(\boldsymbol{y}) \\ g_j^k(\boldsymbol{y}) & \geq & -g_i^k(\boldsymbol{y}) \end{array}\right\} \quad i = 1, \ldots, NI_C^k, \; i \neq j \tag{3.82}$$

and (3.80) becomes

$$\mu_B \geq L_0^k(\boldsymbol{y}, \lambda^k, \mu^k) - NI_C^k \cdot g_j^k(\boldsymbol{y}).$$

(b) If $g_j^k(\boldsymbol{y}) \leq 0$, then $|g_j^k(\boldsymbol{y})| = -g_j^k(\boldsymbol{y})$, and (3.81) reduces to the two inequalities

$$\left.\begin{array}{rcl} g_j^k(\boldsymbol{y}) & \leq & g_i^k(\boldsymbol{y}) \\ g_j^k(\boldsymbol{y}) & \leq & -g_i^k(\boldsymbol{y}) \end{array}\right\} \quad i = 1, \ldots, NI_C^k, \; i \neq j \tag{3.83}$$

and (3.80) becomes

$$\mu_B \geq L_0^k(\boldsymbol{y}, \lambda^k, \mu^k) + NI_C^k \cdot g_j^k(\boldsymbol{y}).$$

The two cases presented above indicate how the summation in (3.79) can be replaced by a linear term when $g_j^k(\boldsymbol{y})$ represents the maximum of all the *qualifying* constraints at a given value of $\boldsymbol{y}$. This concept can then be extended to cover the entire region for $\boldsymbol{y}$. To do this, the above procedure needs to be repeated for all values of j, resulting in $2 \times NI_C^k$ subproblems that need to be solved in order to properly underestimate the Lagrange function at all values of $\boldsymbol{y}$.

Remark 3.14.1 It should be noted that with the use of the linear branching scheme, the same space in $\boldsymbol{y}$ is now spanned by a linear number of underestimators (as opposed to an exponential number in the original algorithm). Therefore, these underestimators will be less tight, and at the end of each iteration, the lower bounds obtained from the dual problems with the linear branching scheme will be looser than those obtained with the original algorithm, resulting in an increase in the number of iterations required for convergence. At the same time, the number of subproblems solved at each iteration is vastly reduced. Therefore, the total computational effort required for the entire algorithm is likely to be much smaller with the linear branching scheme.

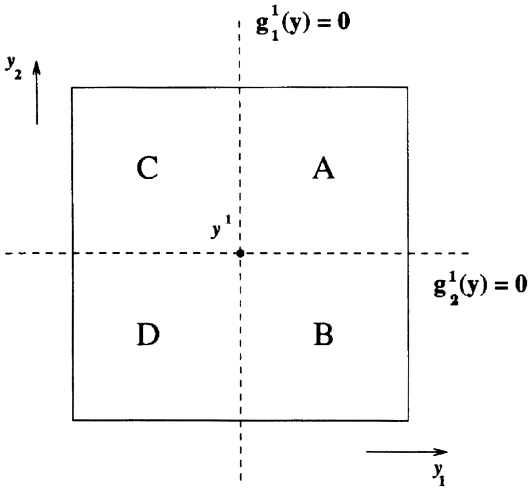
3.14.2 Illustration

Consider the following problem:

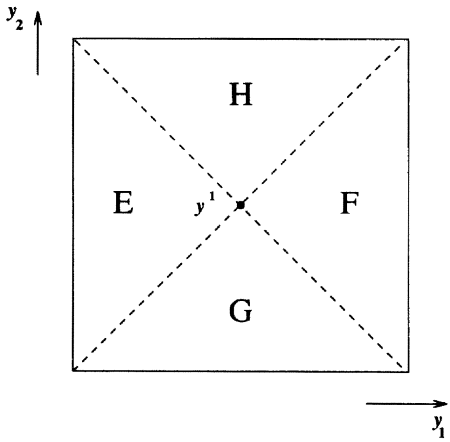
$$\begin{array}{ll} \min\limits_{x,y} & -x_1y_1 - x_2y_2 \\ s.t. & x_1 - y_1 = 0 \\ & x_2 - y_2 = 0 \\ & -1 \leq x, y \leq 1. \end{array}$$

Suppose that the GOP algorithm is applied to this problem, with the starting point of $y = 0$. The first primal problem has the solution $x = 0$, $\lambda_1^1 = 0$ and $\lambda_2^1 = 0$. This leads to the following constraint in the first relaxed dual problem:

$$\begin{array}{rcl} \mu_B & \geq & x_1(0 - y_1) + x_2(0 - y_2) \\ & \geq & -|0 - y_1| - |0 - y_2| \end{array}$$



(a) Original *qualifying* constraints



(b) Transformed *qualifying* constraints

Figure 3.12. Transformed qualifying constraints

where $g_1^1(\boldsymbol{y}) = 0 - y_1$ and $g_2^1(\boldsymbol{y}) = 0 - y_2$ are the two *qualifying* constraints. The region in the $\boldsymbol{y}$ variables, as well as its division using these *qualifying* constraints as used by the original GOP and GOP/MILP algorithms, is shown in Figure 3.12(a). Note that the four regions A, B, C and D represent the four relaxed dual subproblems solved by the original algorithms.

Suppose that $|g_1^1(\boldsymbol{y})| \geq |g_2^1(\boldsymbol{y})|$. There are two possibilities:

(a) $g_1^1(\boldsymbol{y}) \geq 0$. Then, the use of (3.82) results in

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

The region of $\boldsymbol{y}$ described by these constraints is shown as region E in Figure 3.12(b). The corresponding constraint for the relaxed dual problem is given by

$$\mu_B \geq -2g_1^1(\boldsymbol{y}) = 2y_1.$$

(b) $g_1^1(\boldsymbol{y}) \leq 0$. Then, the use of (3.83) results in

$$\begin{aligned} y_1 - y_2 &\geq 0 \\ y_1 + y_2 &\geq 0 \\ y_1 &\geq 0. \end{aligned}$$

These equations describe region F in Figure 3.12(b). The corresponding constraint for the relaxed dual problem is given by

$$\mu_B \geq 2g_1^1(\boldsymbol{y}) = -2y_1.$$

Similarly, when $|g_2^1(\boldsymbol{y})| \geq |g_1^1(\boldsymbol{y})|$, there are two possibilities:

(a) $g_2^1(\boldsymbol{y}) \geq 0$. Then, the use of (3.82) results in

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

The region of $\boldsymbol{y}$ described by these constraints is shown as region G in Figure 3.12(b). The corresponding constraint for the relaxed dual problem is given by

$$\mu_B \geq -2g_2^1(\boldsymbol{y}) = 2y_2.$$

(b) $g_2^1(\boldsymbol{y}) \leq 0$. Then, the use of (3.83) results in

$$\begin{aligned} y_2 - y_1 &\geq 0 \\ y_2 + y_1 &\geq 0 \\ y_2 &\geq 0. \end{aligned}$$

These equations describe region H in Figure 3.12(b). The corresponding constraint for the relaxed dual problem is given by

$$\mu_B \geq 2g_1^1(\boldsymbol{y}) = -2y_1.$$

Thus, it can be seen that the use of the equations (3.82) and (3.83) result in a new set of partitions of the region in $\boldsymbol{y}$. For this example, there are still 4 partitions, so there is no reduction in the number of subproblems to be solved. However, when the number of *connected* variables is more than 2, the use of these transformations will result in a linearly increasing (as opposed to exponentially increasing) number of subproblems at each iteration. For example, when there are 10 *connected* variables, the new partitioning scheme requires 20 relaxed dual subproblems as opposed to 1024 for the original GOP algorithm.

Summary

This chapter has focussed on the theoretical developments of the GOP Algorithm proposed and studied extensively by Floudas and Visweswaran (1990, 1993) and Visweswaran and Floudas (1992, 1993, 1996a). These developments are based on duality theory and on decomposition principles and are applicable to biconvex and bilinear problems, problems with quadratic objective and/or quadratic constraints, as well as certain classes of polynomial problems. The GOP approach has been presented via a cutting plane framework and via a branch and bound framework.

4 THE GOP APPROACH : IMPLEMENTATION AND COMPUTATIONAL STUDIES

This chapter presents the implementation of the GOP algorithm and a variety of computational studies which include problems in heat exchanger network synthesis, pooling and blending, separation networks, as well as concave and indefinite quadratic problems studied by Visweswaran and Floudas (1996b), (1990), (1993). Section 4.1 discusses the various aspects involved in the implementation, including reduction tests and local enhancements at each node of the tree. In particular, the movement of data from one part of the program to another is discussed in detail. Section 4.2 focuses on the computational studies.

4.1 IMPLEMENTATION OF THE GOP AND GOP/MILP ALGORITHMS

In Chapter 3 (sections 3.12 and 3.13) we discussed the branch and bound framework for the GOP and GOP/MILP algorithms proposed by Visweswaran and Floudas (1996a). These algorithms have been implemented in a complete package *cGOP* (Visweswaran and Floudas, 1995). This section describes the key features of the implementation.

4.1.1 The **cGOP** *package*

The *cGOP* package is written entirely in the C programming language. The algorithms can be called either in standalone mode or as subroutines from within another program. The primal and relaxed dual subproblems are solved

C. A. Floudas, *Deterministic Global Optimization*
© Springer Science+Business Media Dordrecht 2000

either using CPLEX (for linear or mixed integer linear problems) or MINOS 5.4 for nonlinear problems. Various options are available to change the routines that are used, such as obtaining tighter bounds on the $\boldsymbol{x}$ variables and $g_i^k(\boldsymbol{y})$ (the gradients of the Lagrange function), as well as solving the full problem as a local optimization problem at each node.

4.1.1.1 Data Structures. The primary data structures are used to describe the problem data, the solutions of the various primal problems, the data for the various Lagrange functions, and the solutions of the relaxed dual subproblems at each iteration.

To facilitate easy and general use of the problem data, the implementation was written assuming that the following types of problems would be solved:

$$\begin{array}{lll} \min\limits_{\boldsymbol{x},\boldsymbol{y}} & \boldsymbol{c}_0^T\boldsymbol{x} + \boldsymbol{d}_0^T\boldsymbol{y} + \boldsymbol{x}^T\boldsymbol{Q}_0\boldsymbol{y} + F_0(\boldsymbol{x}) + G_0(\boldsymbol{y}) & \\ s.t. & l_j \le \boldsymbol{c}_j^T\boldsymbol{x} + \boldsymbol{d}_j^T\boldsymbol{y} + \boldsymbol{x}^T\boldsymbol{Q}_j\boldsymbol{y} \le u_j, & i = 1,\ldots,M_1 \\ & F_j(\boldsymbol{x}) + G_j(\boldsymbol{y}) \le u_j & j = M_1+1,\ldots,M_2 \\ & L \le \begin{pmatrix} \boldsymbol{x} \\ \boldsymbol{y} \end{pmatrix} \le U & \end{array} \tag{4.1}$$

where $j = 1,\ldots,M_1$ are the set of bilinear constraints, and $j = M_1+1,\ldots,M_2$ are the set of general nonlinear constraints. It is assumed that the functions $F_j(\boldsymbol{x})$ and $G_j(\boldsymbol{y})$ are convex in $\boldsymbol{x}$ and $\boldsymbol{y}$ respectively. Note that while the bilinear constraints can be equalities or inequalities, the other nonlinear terms in the constraints are assumed to lie in convex inequalities.

Given the formulation (4.1), the data for the problem can be separated into one part containing the linear and bilinear terms, and another part containing the nonlinear terms $F_j(\boldsymbol{x})$ and $G_j(\boldsymbol{y})$. The first part can be specified through a data file or as arguments during the subroutine call that runs the algorithm. The nonlinear terms, which in general cannot be specified using data files, can be given through user defined subroutines that compute the contribution to the objective function and constraints from these terms, as well as their contribution to the Hessian of the objective function and the Jacobian of the constraints. The problem data is therefore carried in one data structure (called *pdat* from here on, and shown in Figure 4.1) that describes the following items:

Control Data This refers to the type of the problem (bilinear, quadratic, nonlinear), number of $\boldsymbol{x}$ and $\boldsymbol{y}$ variables, the number of constraints, type and value of the starting point for the $\boldsymbol{y}$ variables, as well as tolerances for convergence.

Bilinear Data For reasons of convenience, the linear and bilinear terms in the objective function and constraints are treated together. The data is stored in sparse form, with only the nonzero terms being stored. For each term, the value of the term as well as the indices of its $\boldsymbol{x}$ and/or $\boldsymbol{y}$ terms are stored.

Bounds The global bounds on the variables (which can be changed before the start of the algorithm, but thereafter remain constant) are stored in arrays.

Nonlinear Data The pointers to the functions that compute the nonlinear terms and their gradients are stored in the data structure.

Iteration Data Various counters and loop variables that control and aid in the progress of the iterations are stored in the main data structure. In addition, the best solution obtained by the algorithm so far is also stored.

It is important to note that almost all of the main data structure, once it has been read in from the data file or passed to the main subroutine in the algorithm, remains constant throughout the progress of the algorithm. The only exceptions are the iteration variables and the best solution obtained by the algorithm so far.

The solution of the primal problem is stored together as another data structure, *psol* (shown in Figure 4.2). This contains the value of $\boldsymbol{y}^K$ for which the primal problem was solved, the value of the $\boldsymbol{x}$ variables at solution, the marginals for all the constraints and variables at their bounds, the value of the objective function, as well as an indicator of whether the primal was feasible or not.

Because of the form (4.1), the Lagrange function (for iterations with feasible primal problems) can be written (after linearization of the terms with respect to $\boldsymbol{x}$ and substitution of the KKT optimality conditions for the primal problem) as

$$L(\boldsymbol{x}, \boldsymbol{y}, \lambda^K)\big|_{\boldsymbol{x}^K}^{lin} = L_C + L_L^T \boldsymbol{y} + \sum_{i=1}^{NI_c^K} x_i g_i^T(\boldsymbol{y} - \boldsymbol{y}^K) + G'(\boldsymbol{y})$$

where $G'(\boldsymbol{y})$ represents all the nonlinear terms weighted by the marginals, and can be written as

$$G'(\boldsymbol{y}) = G_0(\boldsymbol{y}) + \sum_{j=M_1+1}^{M_2} \lambda_j^K G_j(\boldsymbol{y}).$$

By introducing new variables to represent the nonlinear constraints, the Lagrange function can be rewritten as

$$L(\boldsymbol{x}, \boldsymbol{y}, \lambda^K)\big|_{\boldsymbol{x}^K}^{lin} = L_C + L_L^T \boldsymbol{y} + \sum_{i=1}^{NI_c^K} x_i g_i^T(\boldsymbol{y} - \boldsymbol{y}^K) + \sum_{j=M_1+1}^{M_2} \lambda_j^K z_j \quad (4.2)$$

$$z_j \geq G_0(\boldsymbol{y}) + G_j(\boldsymbol{y}). \quad (4.3)$$

Note that a simplistic implementation of the algorithm for the general nonlinear problem in (4.1) leads to a problem with nonlinear terms in each Lagrange function, making it much more computationally intensive. Given the fact that the nonlinear terms are the same in each Lagrange function except for a factor

```
struct pdat {
  /* Control section */
  char     *probname;    /* Name of original problem */
  char     objtype;      /* Type of objective function */
  char     contype;      /* Type of constraints */
  char     primaltype;   /* Type of primal problems */
  char     rdualtype;    /* Type of relaxed dual problems */
  int      nxvar;        /* Number of x variables */
  int      nyvar;        /* Number of y variables */
  int      ncon;         /* Number of constraints */
  int      nzcnt;        /* Total number of non-zeros */

  /* Data */
  char     *ctype;       /* Type of X and Y variables */
  int      *sense;       /* Sense of row:  <=, ==, >= */
  double   *rhs;         /* Right hand sides of the rows */
  int      *count;       /* Number of entries in each row */
  int      *begin;       /* Start of entries for each row */
  TERMS    terms;        /* Bilinear terms in problem */
  double   *xlbd, *xubd; /* Bounds on X variables */
  double   *ylbd, *yubd; /* Bounds on Y variables */
  double   objconst;     /* Constants in the objective */
  double   epsa;         /* Absolute tolerance specified */
  double   epsr;         /* Relative tolerance specified */
  int      maxiter;      /* Maximum number of iterations */

  /* Various functions */
  void     userobj();    /* Nonlinear terms in objective */
  void     usercon();    /* Nonlinear terms in constraints */

  /* Solution */
  int      niter;        /* Number of iterations so far */
  double   primalubd;    /* Current upper bound from primals */
  double   rdlbd;        /* Current lower bound from duals */
  double   *x;           /* Starting point, solution for X */
  double   *y;           /* Starting point, solution for Y */
  double   abserror;     /* Absolute error between bounds */
  double   relerror;     /* Relative error between bounds */
};
```

Figure 4.1. Main data structure for the GOP and GOP/MILP algorithms

```
struct psol {
  int      modstat;      /* Feasible or infeasible */
  int      nxvar;        /* Number of x variables  */
  int      nyvar;        /* Number of y variables  */
  int      ncon;         /* Number of constraints */
  double   *yval;        /* Fixed values for Y variables */
  double   objval;       /* Objective value for primal */
  double   *varval;      /* Solution for X variables */
  double   *cmargval;    /* Marginals for constraints */
  double   *bmargval;    /* Marginals for bounds */
  char     *varstat;     /* Status for each variable */
  char     *solver;      /* Which solver was used */
};
```

Figure 4.2. Solution of the Primal Problem

```
/* Structure to hold the data for the Lagrange function */
typedef struct lagdata {
  int      NIc;          /* Number of connected X */
  int      nyvar;        /* Number of Y variables */
  double   *xlbd;        /* Lower bounds for connected X */
  double   *xubd;        /* Lower bounds for connected X */
  int      *xindex;      /* Indices of connected X */
  double   *ylbd, *yubd; /* Bounds on Y variables */
  double   *glbd, *gubd; /* Bounds on qualifying constraints */
  double   **glin;       /* Terms in qualifying constraints */
  double   *gconst;      /* Constants in qualifying const. */
  double   *llin;        /* Terms in Lagrange function */
  double   lconst;       /* Constants in Lagrange function */
};
```

Figure 4.3. Lagrange function data structure

due to the marginals λ_j^K, it is far more efficient to group the terms together, and therefore to compute their gradients only once. Moreover, the regrouping of the terms means that as far as each individual Lagrange function is concerned, only the data regarding (4.2) need to be stored, that is, the coefficients of the linear terms L_L^T, the coefficients of the bilinear terms g_i^T and the multipliers λ_j^K. Its structure is shown in Figure 4.3.

The solutions of the relaxed dual subproblems comprise the last major data structure. Apart from the actual objective value for the solution and the values of the y variables, this data includes information about which iteration and

parent node generated each child node in the branch and bound tree. Thus, the entire information about the tree is stored in the array of relaxed dual solution structures, *rdsol*.

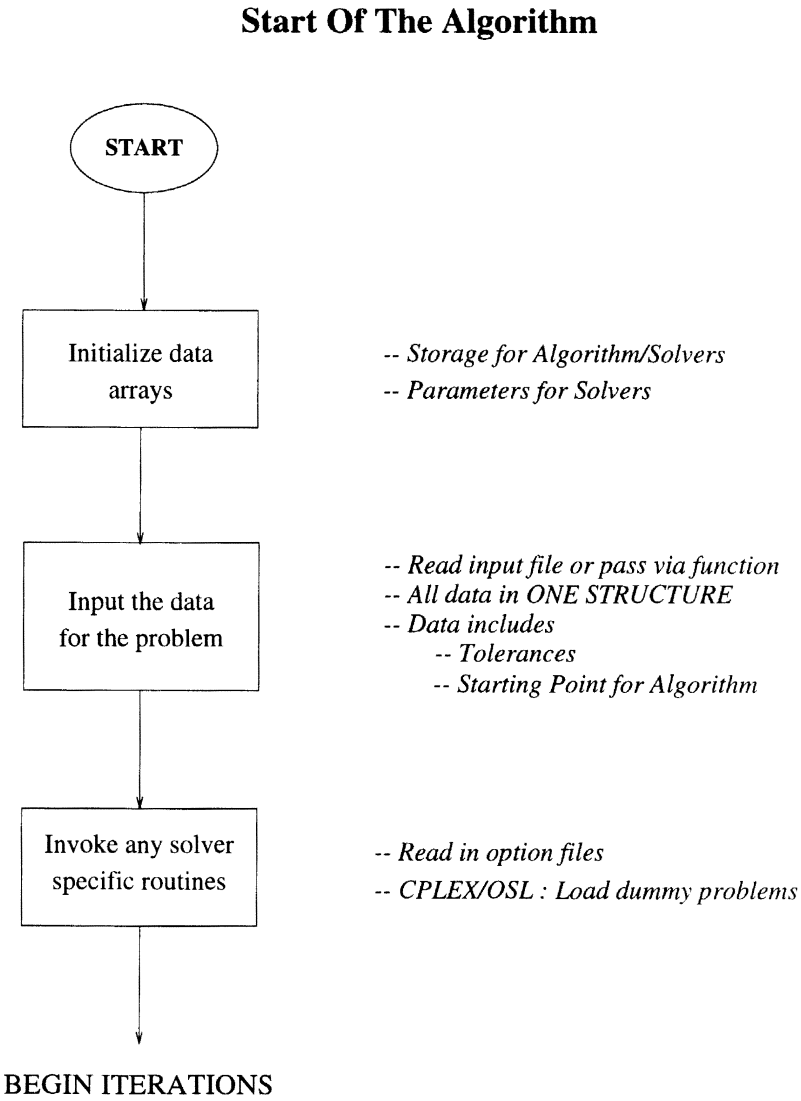


Figure 4.4. Implementation of the GOP Algorithm in C

Based upon these various data units, the overall scheme of the implementation is now presented. A pictorial view of the algorithm is given in Figure (4.4).

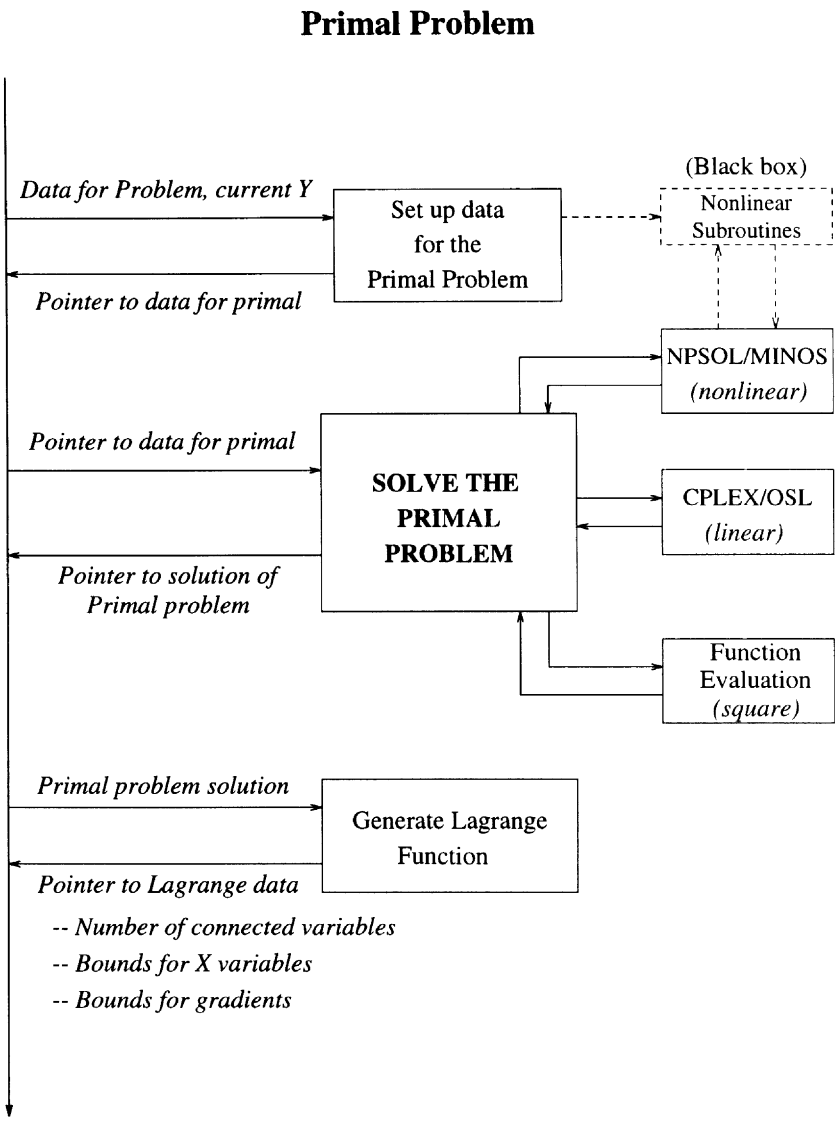


Figure 4.4 *(continued).* Implementation of the GOP Algorithm in C

4.1.1.2 Initialization of parameters. At the start of the algorithm, the list of relaxed dual solutions *rdsol* is initialized to contain the starting point for the y variables, indicating the root node for the whole branch and bound tree. An initial local optimization problem can be solved to find a good upper bound and starting point for the y variables, if desired. Various counters and bookkeeping variables are initialized before the start of the iterations.

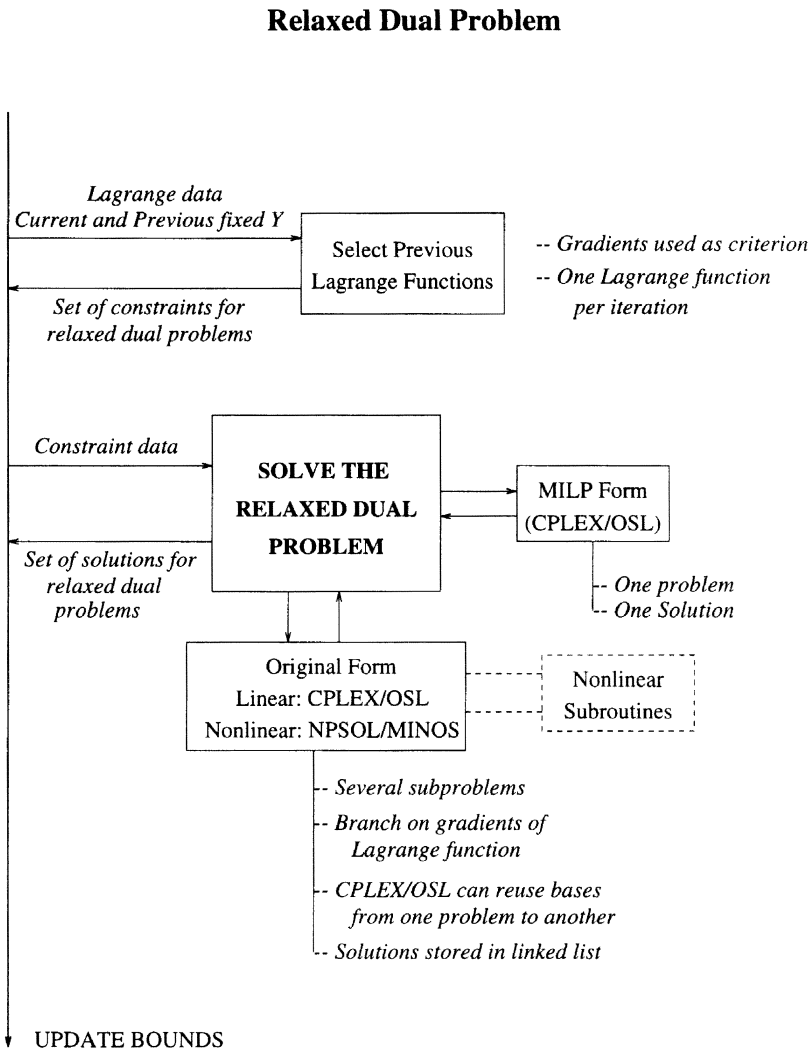


Figure 4.4 *(continued).* Implementation of the GOP Algorithm in C

4.1.1.3 Selection Of Previous Lagrange Functions and Current Region. At any given iteration, the relaxed dual subproblems will contain a Lagrange function from the current iteration, and one from each of the parent nodes of the current node in the branch and bound tree. In order to select these functions, a backward search is done through the list of solutions to the relaxed dual problems starting from the current node (i.e., the node that has been chosen at the end of the previous iteration). The following steps are repeated:

Selecting The Best Solution and Lower Bound

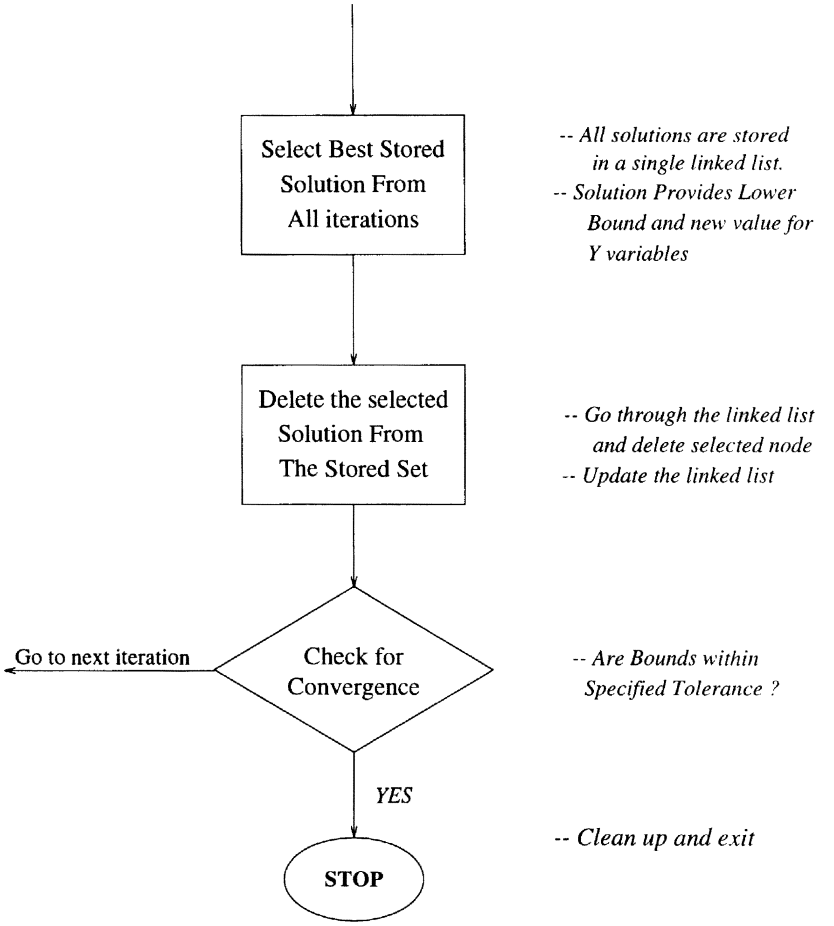


Figure 4.4 *(continued).* Implementation of the GOP Algorithm in C

Step 0. Initialize lagsel[MAXITER], the array of parent nodes for the current node.

Step 1. Add the current node C to lagsel. Set lagsel[1] $= C$, and set the number of Lagrange functions $numlag = 1$.

Step 2. Find the iteration P that generated the current node.

Step 3. Go to the node corresponding to iteration P (say node D) and add this node to the list, i.e. set $numlag = numlag + 1$, lagsel[$numlag$] $= D$.

Step 4. Repeat Steps 2 and 3 until the root node has been reached.

The list of nodes generated in the above steps provides a set of *qualifying* constraints (one set per node) that define the region of operation for the current node.

4.1.1.4 Obtaining Tighter Bounds For The x Variables. If desired, a set of problems are solved that try to find the tightest bounds on the x and y variables given any linear and convex constraints in the original problem, and the current region for the y variables as defined by the *qualifying* constraints for the parent nodes of the current node. This is a very important step, because the tightness of the bounds on the x variables is crucial to obtaining tight underestimators for the relaxed dual problems.

4.1.1.5 Primal problem. The primal problem takes as data the *pdat* structure, along with the current vector for y^K. It is also given the current region for the problem as defined by the selected *qualifying* constraints. There are several schemes that can be followed to solve the primal problem, all of which involve various combinations of the primal, relaxed primal or a local optimization problem solved in the current region. One possible scheme is as follows:

1. Solve the primal problem at the current y^K.

2. If the primal problem is feasible, update the upper bound.

   (a) Solve the full NLP as a local optimization problem in the current region.

   (b) If the NLP solution is lower than the upper bound, replace y^K with the NLP solution and go to Step 1. Otherwise go to Step 4.

3. If the primal problem is infeasible

   (a) Solve the full NLP as a local optimization problem in the current region.

   (b) If the NLP provides a feasible solution, then replace y^K with the new solution from the NLP and go to Step 1. Otherwise, solve the relaxed primal problem go to Step 4.

4. Return the solution of the problem as a *psol* data structure.

4.1.1.6 Determination of Connected Variables. The solution of the current primal (or relaxed primal) problem is used to determine the set of *connected* variables. Several reduction tests are used to determine the set. These include testing for the lower and upper bounds on the gradients of the Lagrange function and the tightness of the bounds on the x variables. If the lower and upper bounds on an x variable are within a certain tolerance, that variable can be fixed at its bound. Provision is also made for user defined tests for reducing the number of *connected* variables.

4.1.1.7 Generation of Lagrange Function Data. As mentioned earlier, only the data for the Lagrange functions (4.2) are stored. This data is generated from the current *psol* structure. Once the data is generated, it can be used again whenever the Lagrange functions from that iteration need to be generated.

4.1.1.8 Global Lagrange Functions. If there are no connected variables in the Lagrange function generated at the current iteration, then this function contains only the $\boldsymbol{y}$ variables. Therefore, it is a valid underestimator for the entire $\boldsymbol{y}$ space, and can be included as a cut for all future relaxed dual subproblems. In such a case, the current Lagrange function is added to the list of "global" Lagrange functions.

4.1.1.9 Relaxed Dual Problem. Given the current region and a set of *connected* variables, the region is partitioned using the *qualifying* constraints of the current Lagrange function. Then, a relaxed dual subproblem is solved in each region, and the solutions are stored as part of *rdsol* if feasible. The nonlinear terms in the objective function and constraints are again incorporated through calls to the user defined functions. In the case of the GOP/MILP algorithm, only one MILP problem needs to be solved.

4.1.1.10 Selection of the Lower Bound. After the relaxed dual problem has been solved for every possible combination of the bounds of the connected variables (in the case of the GOP/MILP algorithm, after the MILP has been solved), a new lower bound needs to be determined for the global solution. Since the solutions are all stored as a linked list, this permits a simple search for the best solution. This solution is then removed by simply deleting the corresponding node from the linked list. At the same time, the corresponding value of $\boldsymbol{y}$ is also extracted to use for the next iteration.

4.1.1.11 Resolving the MILP Formulation. For the GOP/MILP formulation, after a solution has been selected from the list of candidate solutions, the MILP formulation corresponding to the iteration from which the solution was generated needs to be resolved. To accomplish this, a binary cut that excludes the selected solution is generated and added to the MILP formulation, which is then solved. Because of the likelihood that the formulation for any given iteration is likely to be solved at least a few times, several such formulations are stored in memory, so that when they are resolved, it is merely a matter of restarting the problem with the additional binary cut. This saves valuable loading and startup time for the solution of these problems.

4.1.1.12 Convergence. Finally, the check for convergence is done. The algorithm converges if the relative difference between the upper bound from the primal problems and the lower bound from the relaxed dual problems is less than ϵ. Then, the algorithm terminates (in the case of the standalone version) or returns to the calling routine (in case of the subroutine version).

Otherwise, the algorithm continues with the new fixed value of $\boldsymbol{y}$ for the primal problem found from the previous step.

4.2 COMPUTATIONAL STUDIES

In this section, we present the results of the application of the *cGOP* package to various problems in chemical engineering design and mathematical programming. Additional computational studies, including studies based on other implementations of the GOP can be found in Visweswaran and Floudas (1996b), (1990), (1992), (1993).

4.2.1 Heat Exchanger Network Problems

Heat exchanger network synthesis problems have traditionally been solved using a decomposition strategy, where the aims of targeting, selection of matches and optimization of the resulting network configuration are treated as independent problems. Given the minimum utility requirements and a set of matches, a superstructure of all the possible alternatives is formulated (see Floudas (1995)). The resulting optimization problem is nonconvex. In this section, two such superstructures of heat exchanger networks are solved using the GOP algorithm.

The problems solved in this section have the following form:

$$\text{OBJ} \quad = \quad \min \sum_{ij \in MA} \alpha_{ij} \left(\frac{Q_{ij}}{U_{ij} LMTD_{ij}}\right)^{\beta_{ij}}$$

s.t.

(Initial splitter mass balance)

$$\sum_{k' \in R_k} f^I_{k'} \;=\; F^k$$

(Mixer balances at exchanger inlets)

$$f^I_{k'} + \sum_{k'' \in S_{k'}} f^B_{k',k''} - f^E_{k'} \;=\; 0 \quad \forall\, k \in HCT.$$

(Splitter balances at exchanger outlets)

$$f^O_{k'} + \sum_{k'' \in S_{k'}} f^B_{k'',k'} - f^E_{k'} \;=\; 0 \quad \forall\, k \in HCT.$$

(Energy balances at mixers)

$$T^k f^I_{k'} + \sum_{k'' \in S_{k'}} f^B_{k',k''} t^O_{k''} - f^E_{k'} t^I_{k'} \;=\; 0 \quad \forall\, k \in HCT.$$

(Energy balances in exchangers)

$$Q_{ij} = f_i^{E,j}(t_i^{I,j} - t_i^{O,j}) \quad \forall\ (ij) \in MA$$

$$Q_{ij} = f_j^{E,i}(t_j^{O,i} - t_j^{O,i}) \quad \forall\ (ij) \in MA$$

$$LMTD_{ij} = \frac{2}{3} \times (DT1_{ij} \times DT2_{ij})^{1/2} + \frac{1}{6} \times (DT1_{ij} + DT2_{ij}).$$

Here, U_{ij} are the fixed heat transfer coefficients. It should be noted that for fixed Q_{ij}, the objective function is convex. Therefore, by *projecting* on the flow rates f_i, the primal problem becomes convex in the remaining variables (the temperatures and temperature differences). Linearization of the Lagrange function ensures that the relaxed dual subproblems are LP subproblems in the flowrates.

Example 4.2.1 This example is taken from Floudas and Ciric (1989). In this problem, the objective is to determine the globally optimal network for a system of two hot streams and one cold stream. The superstructure of all possible solutions is shown in Figure 4.5. Based upon this superstructure, the model can be formulated as the following optimization problem :

$$\min \quad 1300\Big[\frac{1000}{0.05\big[\frac{2}{3}(\Delta T_{11}\Delta T_{12})^{0.5}\big] + \frac{1}{6}(\Delta T_{11} + \Delta T_{12})}\Big]^{0.6} + 1300\Big[\frac{600}{0.05\big[\frac{2}{3}(\Delta T_{21}\Delta T_{22})^{0.5}\big] + \frac{1}{6}(\Delta T_{21} + \Delta T_{22})}\Big]^{0.6}$$

$s.t.$

$$\begin{aligned}
f_1^I + f_2^I &= 10 \\
f_1^I + f_{12}^B - f_1^E &= 0 \\
f_2^I + f_{21}^B - f_2^E &= 0 \\
f_1^O + f_{21}^B - f_1^E &= 0 \\
f_2^O + f_{12}^B - f_2^E &= 0 \\
150 f_1^I + t_2^O f_{12}^B - t_1^I f_1^E &= 0 \\
150 f_2^I + t_1^O f_{21}^B - t_2^I f_2^E &= 0 \\
f_1^E(t_1^O - t_1^I) &= 1000 \\
f_2^E(t_2^O - t_2^I) &= 600 \\
\Delta T_{11} = 500 - t_1^O, \quad \Delta T_{12} &= 250 - t_1^I \\
\Delta T_{21} = 350 - t_2^O, \quad \Delta T_{22} &= 200 - t_2^I \\
\Delta T_{11}, \Delta T_{12}, \Delta T_{21}, \Delta T_{22} &\geq 10 \\
0 \leq f_1^I, f_2^I, f_1^O, f_2^O, f_1^E, f_2^E, f_{12}^B, f_{21}^B &\leq 10.
\end{aligned}$$

Considering the set of possible solutions inherent in Figure 4.5, it is obvious that the bypass streams (f_{12}^B and f_{21}^B) can never be simultaneously active, i.e. at least one of these streams has to be zero. Therefore, two different problems can be solved, one with $f_{12}^B = 0$ and another with $f_{21}^B = 0$. When the GOP algorithm is applied to the problem in this form, the optimal solution (given in Figure 4.6) is found in 11 iterations, needing 0.54 cpu seconds on an HP 730.

Example 4.2.2 This example is also taken from Floudas and Ciric (1989). It features three hot streams and two cold streams.

$$
\begin{aligned}
\min \quad & 1300\Big[\frac{2000}{0.5\big[\frac{2}{3}(\Delta T_{11}\Delta T_{12})^{0.5}\big] + \frac{1}{6}(\Delta T_{11} + \Delta T_{12})}\Big]^{0.6} + \\
& 1300\Big[\frac{1000}{1.0\big[\frac{2}{3}(\Delta T_{21}\Delta T_{22})^{0.5}\big] + \frac{1}{6}(\Delta T_{21} + \Delta T_{22})}\Big]^{0.6} + \\
& 1300\Big[\frac{1500}{2.0\big[\frac{2}{3}(\Delta T_{31}\Delta T_{32})^{0.5}\big] + \frac{1}{6}(\Delta T_{31} + \Delta T_{32})}\Big]^{0.6}
\end{aligned}
$$

$$
\begin{aligned}
s.t. \quad f_1^I + f_2^I + f_3^I &= 45 \\
f_1^I + f_{12}^B + f_{13}^B - f_1^E &= 0 \\
f_2^I + f_{21}^B + f_{23}^B - f_2^E &= 0 \\
f_3^I + f_{31}^B + f_{32}^B - f_3^E &= 0 \\
f_1^O + f_{21}^B + f_{31}^B - f_1^E &= 0 \\
f_2^O + f_{12}^B + f_{33}^B - f_2^E &= 0 \\
f_3^O + f_{13}^B + f_{23}^B - f_3^E &= 0 \\
100 f_1^I + t_2^O f_{12}^B + t_3^O f_{13}^B - t_1^I f_1^E &= 0 \\
100 f_2^I + t_1^O f_{21}^B + t_3^O f_{23}^B - t_2^I f_2^E &= 0 \\
100 f_3^I + t_3^O f_{31}^B + t_2^O f_{32}^B - t_3^I f_3^E &= 0 \\
f_1^E(t_1^O - t_1^I) = 2000, \quad f_2^E(t_2^O - t_2^I) = 1000, \quad f_3^E(t_3^O - t_3^I) &= 1500 \\
\Delta T_{11} = 210 - t_1^O, \quad \Delta T_{21} = 210 - t_2^O, \quad \Delta T_{31} &= 210 - t_3^O \\
\Delta T_{12} = 130 - t_1^I, \quad \Delta T_{22} = 160 - t_2^I, \quad \Delta T_{32} &= 180 - t_3^I \\
\Delta T_{11}, \Delta T_{12}, \Delta T_{21}, \Delta T_{22} \Delta T_{31}, \Delta T_{32} &\geq 10 \\
0 \leq f_1^I, f_2^I, f_3^I, f_1^O, f_2^O, f_3^O, f_{12}^B, f_{13}^B, f_{21}^B, f_{23}^B, f_{31}^B, f_{32}^B &\leq 45.
\end{aligned}
$$

The superstructure for this example is shown in Figure 4.7. In Visweswaran and Floudas (1996b), the flowrate between heat exchanger 1 and heat exchanger 3, was restricted to zero, that is $f_{31}^B = 0$. There are a total of 27 variables and 19 constraints (of which six are bilinear). With a *projection* on the flow rates, there are six connected variables. The GOP algorithm requires a total of 39 iterations and 54.62 cpu seconds to solve this problem. The optimal solution found by the algorithm is given in Figure 4.8.

4.2.2 Heat Exchanger Network Problems With Linear Cost Functions

In this section, we apply the GOP algorithm to the global optimization of several heat exchanger networks with fixed topology. The problems are taken from Quesada and Grossmann (1993) and assume linear cost functionals for the exchanger areas as well as arithmetic mean driving forces for the temperature differences between the exchanging streams. Under these assumptions, the problems reduce to the minimization of a sum of linear fractional functions (which is nonconvex) over a set of linear constraints.

In order to reduce these problems to a form where the GOP algorithm could be applied, we employ the ideas of Liu and Floudas (1993), which involve a difference of convex functions transformation. These ideas formed the basis for the theoretical development of the αBB global optimization approach that we discuss in Part III of this book. This involves first the reduction of the problem to a simpler form, and subsequent use of eigenvalue analysis on the resulting fractional objective functions in order to determine the smallest quadratic terms that are needed to "convexify" the objective function. Since this method is very general and can be of use in various problems of this type, it is outlined in some detail here for one of the examples.

This example (Example 4 of Quesada and Grossmann, 1993) features a network of three exchangers used to heat one cold stream and cool three hot streams. This network is shown in Figure 4.9, with $FC_p = 10$ for all the streams. The minimum temperature of approach is $10^0 K$.

The problem formulation, featuring constraints for the heat balances, minimum temperature approaches and feasibility is shown below:

$$\min \quad \frac{Q_1}{2\Delta T_1} + \frac{Q_2}{2\Delta T_2} + \frac{Q_3}{2\Delta T_3}$$

Temperature Differences:

$$\begin{aligned} 2\Delta T_1 &= 150 + T_1 - T_4 \\ 2\Delta T_2 &= 500 + T_2 - T_4 - T_5 \\ 2\Delta T_3 &= 150 + T_3 - T_5 \end{aligned}$$

Heat Balances:

$$\begin{aligned} Q_1 &= 10(T_4 - 300) = 10(450 - T_1) \\ Q_2 &= 10(T_5 - T_4) = 10(500 - T_2) \\ Q_3 &= 10(400 - T_5) = 10(550 - T_3) \end{aligned}$$

Minimum Temperature Approaches:

$$\begin{aligned} T_1 - 300 &\geq 10 \qquad & 450 - T_4 \geq 10 \\ T_2 - T_4 &\geq 10 \qquad & 500 - T_5 \geq 10 \\ T_3 - T_5 &\geq 10 \end{aligned}$$

Feasibility:

$$T_1, T_2, T_3, T_4, T_5 \geq 0.$$

The three heat balance equations can be used to eliminate three of the variables in the problem. Choosing the intermediate streams T_4 and T_5 as the independent variables leads to

$$\begin{aligned} T_1 &= 750 - T_4 \\ T_2 &= 500 + T_4 - T_5 \\ T_3 &= 150 + T_5. \end{aligned}$$

Using the minimum temperature approaches, tighter bounds on T_4 and T_5 are obtained:

$$\begin{array}{lllllllllll} T_1 & \geq & 310 & \Rightarrow & 750 - T_4 & \geq & 310 & \Rightarrow & T_4 & \leq & 440 \\ T_2 & \geq & 10 + T_4 & \Rightarrow & 500 + T_4 - T_5 & \geq & 10 + T_4 & \Rightarrow & T_5 & \leq & 490. \end{array}$$

Similarly the temperature differences reduce to

$$\begin{aligned} \Delta T_1 &= 450 - T_4 \\ \Delta T_2 &= 500 - T_5 \\ \Delta T_3 &= 150. \end{aligned}$$

Thus, the problem formulation reduces to

$$\min \quad 10000 \left[\frac{T_4 - 300}{450 - T_4} + \frac{T_5 - T_4}{500 - T_5} + \frac{400 - T_5}{150} \right]$$

$$300 \leq T_4, T_5 \leq 400.$$

Consider now the three individual terms inside the parentheses. For the sake of clarity, the factor of 10000 is omitted below.

- The first fractional term is

$$F_1 = \frac{T_4 - 300}{450 - T_4}.$$

The Hessian of F_1 is given by

$$\frac{300}{(450 - T_4)^3}$$

which is always positive, since $T_4 \leq 400$. Therefore, this term is convex for all values of T_4 and T_5.

- The third term

$$F_3 = \frac{400 - T_5}{150},$$

is a linear term and therefore always convex.

- The second term is

$$F_2 = \frac{T_5 - T_4}{500 - T_5}.$$

The Hessian of F_2 is given by

$$H_2 = \begin{bmatrix} 0 & \frac{-1}{y^2} \\ \frac{-1}{y^2} & \frac{2x}{y^3} \end{bmatrix} \tag{4.4}$$

where $x = 500 - T_4$ and $y = 500 - T_5$. The eigenvalues of this Hessian are given by

$$E_1, E_2 = \frac{x \pm \sqrt{x^2 + y^2}}{y^3}.$$

It can be seen that the second eigenvalue (for the negative value of the square root) will *always* be negative. Thus, the Hessian has mixed eigenvalues, indicating that the second term in the objective is nonconvex.

In order to "convexify" this term, a quadratic term in one or more of the variables can be added. Suppose that the term αT_4^2 is added. Then, the term becomes

$$F_2' = \frac{T_5 - T_4}{500 - T_5} + \alpha T_4^2.$$

The Hessian of this term is given by

$$H_2' = \begin{bmatrix} 2\alpha & \frac{-1}{y^2} \\ \frac{-1}{y^2} & \frac{2x}{y^3} \end{bmatrix}$$

where again $x = 500 - T_4$ and $y = 500 - T_5$. The eigenvalues of this Hessian are given by

$$E_1, E_2 = \frac{1}{y^3}\left[x + \alpha y^3 \pm \sqrt{(x - \alpha y^3)^2 + y^2}\right].$$

For the second eigenvalue to be positive for all values of T_4 and T_5, the term in the square brackets must be positive. In other words,

$$x + \alpha y^3 \pm \sqrt{(x - \alpha y^3)^2 + y^2} \geq 0.$$

This leads to the inequality

$$\alpha \geq \frac{1}{4xy}.$$

Since $100 \leq x, y \leq 200$, we obtain

$$\alpha \geq \frac{1}{40000}.$$

Thus, adding the term $\frac{1}{40000}T_4^2$ to F_2 is sufficient to make this term convex. The net result of this is that the objective function can now be written as

$$\min \quad 10000\left[\frac{T_4 - 300}{450 - T_4} + \frac{T_5 - T_4}{500 - T_5} + \frac{400 - T_5}{150} + \frac{T_4^2}{40000}\right] - \frac{T_4^2}{4}$$

where the first term is convex, and the second term is concave. By the addition of an extra variable and renaming all the variables, the problem now becomes

$$\min \quad 10000\left[\frac{y_1 - 300}{450 - y_1} + \frac{y_2 - y_1}{500 - y_2} + \frac{400 - y_2}{150} + \frac{y_1^2}{40000}\right] - 0.25x_1y_1$$

$$\begin{aligned} x_1 - y_1 &= 0 \\ 300 \leq x_1, y_1, y_2 &\leq 400. \end{aligned}$$

Now the problem satisfies the conditions of the GOP algorithm, being a convex problem in $\boldsymbol{y}$ for all fixed $\boldsymbol{x}$ and a linear problem in $\boldsymbol{x}$ for all fixed $\boldsymbol{y}$.

Similar reductions were obtained for all the example problems given in Quesada and Grossmann (1993). The results of applying the GOP algorithm to these problems is given in Table 4.1. Note that in all the cases, the problems reduced to either one or two variable unconstrained problems. Consequently, the subproblems solved by the algorithm are very small in size, as shown in the CPU times taken to converge to the optimum.

4.2.3 Pooling and Blending Problems

Pooling and blending problems are a feature of models for most chemical processes. In particular, for problems relating to refinery and petrochemical processing, it is often necessary to model not only the product flows but the properties of intermediate streams as well. These streams are usually combined in a tank or pool, and the pool is used in downstream processing or blending. The presence of these streams in the model introduces nonlinearities, often in a nonconvex manner. The nonconvexities arise from the interactions between the qualities of the input streams and the blended products.

Traditionally, pooling problems have been solved using successive linear programming (SLP) techniques. The first SLP algorithm (Method of Approximation Programming) was proposed by Griffith and Stewart (1961). Subsequently,

Problem Name	*Problem Size*		*GOP Algorithm*	
	Variables	Constraints	Iterations	CPU (sec)
Example 1	12	13	4	0.09
Example 2	12	13	3	0.06
Example 4	11	9	3	0.10
Example 5	11	9	8	0.20
Example 7	26	30	4	0.11

Table 4.1. Heat Exchanger Network Problems from Quesada and Grossmann (1993).

SLP algorithms have been proposed by Lasdon *et al.* (1979), Palacios-Gomez *et al.* (1982) and Baker and Lasdon (1985) among others. These algorithms have been applied to pooling problems by Haverly (1978) and Lasdon *et al.* (1979). SLP algorithms have the advantage that they can utilize existing LP codes and can handle large scale systems easily. However, to guarantee convergence to the global solution, they require convexity in the objective function and the constraints. For this reason, these methods cannot determine the best solution for all pooling problems.

Various formulations have been proposed for pooling and blending problems. In the following sections, we consider the application of the GOP algorithm to three of these formulations, namely, the Haverly Pooling problem, two pooling problems from Ben-Tal and Gershovitz (1994), and a multiperiod tankage quality problem commonly occuring in refineries.

4.2.3.1 The Haverly Pooling Problem. In his studies of the recursive behavior of linear programming *(LP)* models, Haverly (1978) defined a pooling problem as shown in Figure 4.10. Three substances A, B and C with different sulfur contents are to be combined to form two products x and y with specified maximum sulfur contents. In the absence of a pooling restriction, the problem can be formulated and solved as an LP. However, when the streams need to be pooled (as, for example, when there is only one tank to store A and B), the LP must be modified. Haverly has shown that without the explicit incorporation of the effect of the economics associated with the sulfur constraints on the feed selection process, a recursive algorithm for solving a simple formulation having only a pool balance cannot find the global solution. Lasdon *et al.* (1979) added a pool quality constraint to the formulation. This complete NLP formulation is shown below :

Case	Bounds		Cost	Optimal Solution		GOP		GOP/MILP	
	x^U	y^U	of B	$f^\star$	$p^\star$	Iter.	CPU	Iter.	CPU
I	100	200	\$16	-\$400	1.0	12	0.22	12	0.49
II	600	200	\$16	-\$600	3.0	12	0.21	12	0.45
III	100	200	\$13	-\$750	1.5	14	0.26	14	0.56

Table 4.2. Data and results for the Haverly Pooling Problem

$$\min \quad 6A + 16B + 10(C_x + C_y) - \ 9x - 15y$$

$s.t.$

$$P_x + P_y - A - B = 0 \ \} \qquad \textit{pool balance}$$

$$\left.\begin{array}{rcl} x - P_x - C_x & = & 0 \\ y - P_y - C_y & = & 0 \end{array}\right\} \qquad \textit{component balance}$$

$$p.(P_x + P_y) - 3A - B = 0 \ \} \qquad \textit{pool quality}$$

$$\left.\begin{array}{rcl} p.P_x + 2.C_x - 2.5x & \leq & 0 \\ p.P_y + 2.C_y - 1.5y & \leq & 0 \end{array}\right\} \qquad \textit{product quality constraints}$$

$$\left.\begin{array}{rcl} x & \leq & x^U \\ y & \leq & y^U \end{array}\right\} \qquad \textit{upper bounds on products}$$

where p is the sulfur quality of the pool; its lower and upper bounds are 1 and 3 respectively. This problem was solved by both Haverly (1979) and Lasdon *et al.* (1979). In all cases, however, the global optimum could not always be determined, the final solution being dependent on the starting point.

Floudas and Aggarwal (1990) solved the problem using the Global Optimum Search, *GOS*, approach (Floudas *et al.*, 1989). They had to reformulate the problem by adding variables and constraints, and despite being they were successful in finding the global minimum from 28 out of 30 starting points, they could not mathematically guarantee that the algorithm would converge to the global minimum.

The GOP Algorithm By *projecting* on the pooling quality p, the problem becomes linear in the remaining variables. Hence, p is chosen as the "y" variable. From the constraint set, it can be seen that only P_x and P_y are the *connected* variables. Hence, four relaxed dual subproblems need to be solved at each iteration. Three cases of the pooling problem have been solved using the GOP and GOP/MILP algorithms. The data for these three cases, as well as the average number of iterations required by the algorithms to converge, are

given in Table 4.2. It can be seen that in all cases, the algorithms require less than 15 iterations to identify and converge to the global solution.

4.2.3.2 Pooling Problems from Literature. Here we study the application of the GOP algorithm to two pooling problems taken from (Ben-Tal *et al.*, 1994). The following notation is used for these problem models :

$$\begin{aligned}
\{1,2,\cdots,i,\cdots,I\} &\equiv \text{set of components} \\
\{1,2,\cdots,j,\cdots,J\} &\equiv \text{set of products} \\
\{1,2,\cdots,k,\cdots,K\} &\equiv \text{set of qualities} \\
\{1,2,\cdots,l,\cdots,L\} &\equiv \text{set of pools}
\end{aligned}$$

The following variable sets are present in the model :

$$\begin{aligned}
x_{il} &- \text{amount of component } i \text{ allocated to pool } l \\
y_{lj} &- \text{amount going from pool } l \text{ to product } j \\
z_{ij} &- \text{amount of component } i \text{ going to product } j \\
p_{lk} &- \text{level of quality } k \text{ in pool } l
\end{aligned}$$

The parameters in the problem are :

$$\begin{aligned}
A_i &- \text{Upper bounds for component availabilities} \\
D_j &- \text{Upper bounds for product demands} \\
S_l &- \text{Upper bounds for pool sizes} \\
Q_{jk} &- \text{Upper bounds for product qualities} \\
q_{ik} &- \text{Level of quality } k \text{ in component } i \\
c_i &- \text{Unit price of component } i \\
d_j &- \text{Unit price of product } j
\end{aligned}$$

Using this notation, these pooling problems have the following form:

$$\max \quad -\sum_i \sum_l c_i x_{il} + \sum_l \sum_j d_j y_{lj} + \sum_i \sum_j (d_j - c_i) z_{ij}$$

Problem No.	*Problem Size*				*GOP Algorithm*	
	I	J	K	L	Iterations	CPU (HP730)
1.	4	2	1	1	7	0.95
2.	5	5	2	1	41	5.80

Table 4.3. Pooling Problems From Ben-Tal et al. (1994).

$$
\begin{aligned}
s.t. \quad \sum_l x_{il} + \sum_j z_{ij} &\leq A_i \\
\sum_i x_{il} + \sum_j y_{lj} &= 0 \\
\sum_i x_{il} &\leq S_l \\
-\sum_i q_{ik} x_{il} + p_{lk} \sum_j y_{lj} &= 0 \\
\sum_l y_{lj} + \sum_i z_{ij} &\leq D_j \\
\sum_l (p_{lk} - Q_{jk}) y_{lj} + \sum_i (q_{ik} - Q_{jk}) z_{ij} &\leq 0.
\end{aligned}
\tag{4.5}
$$

The data for these problems can be found in Ben-Tal and Gershovitz (1992). The results of application of the GOP algorithm to these problems are shown in Table 4.3.

4.2.3.3 Multiperiod Tankage Quality Problem. We studied this example in Chapter 3 (section 3.11) where the focus was on the effect of the additional properties of the Lagrange function. Here, we study further the application of the GOP and the GOP/MILP algorithms as they are implemented in the *cGOP* package. The complete mathematical formulation for this problem, consists of 39 variables and 22 inequality constraints (of which 12 are nonconvex). The results of the application of the branch-and-bound framework for the GOP and GOP/MILP algorithms are shown in Table 4.4. Note that both algorithmic approaches can solve the multiperiod tankage problem very efficiently.

4.2.4 Problems in Separation Sequences

As in the case of heat exchanger networks, problems involving separations (sharp and nonsharp) can often be posed as a superstructure from which the best alternative is to be selected. The example problem presented here involves

Starting Point	*Original GOP*			*GOP/MILP*	
(y)	Iter.	Subproblems	CPU	Iter	CPU
Lower bound	8	18	3.66	7	14.7
Upper bound	9	19	3.68	9	13.1
$q_{t1} = 100, q_{t2} = 70$	11	18	3.95	13	22.4
$q_{t1} = 80, q_{t2} = 100$	9	19	3.23	13	16.5

Table 4.4. Multiperiod Tankage Quality Problem

the separation of a three component mixture into two multicomponent products using separators, splitters, blenders and pools. The superstructure for the problem (Floudas and Aggarwal, 1990) is given in Figure 4.11. The NLP formulation for the problem is given below:

$$\min \quad 0.9979 + 0.00432F_5 + 0.01517F_{13}$$

subject to

(Overall Mass Balances)

$$\begin{aligned} F_1 + F_2 + F_3 + F_4 &= 300 \\ F_6 - F_7 - F_8 &= 0 \\ F_9 - F_{10} - F_{11} - F_{12} &= 0 \\ F_{14} - F_{15} - F_{16} - F_{17} &= 0 \\ F_{18} - F_{19} - F_{20} &= 0 \end{aligned}$$

(Splitter Component Balances)

$$\begin{aligned} F_5 x_{j,5} - F_6 x_{j,6} - F_9 x_{j,9} &= 0 \quad j = A, B, C \\ F_{13} x_{j,13} - F_{14} x_{j,14} - F_{18} x_{j,18} &= 0 \quad j = A, B, C \end{aligned}$$

(Inlet Mixer Balances)

$$\begin{aligned} 0.333F_1 + F_{15} x_{j,14} - F_5 x_{j,5} &= 0 \quad j = A, B, C \\ 0.333F_2 + F_{10} x_{j,9} - F_{13} x_{j,13} &= 0 \quad j = A, B, C \\ 0.333F_3 + F_7 x_{A,6} + F_{11} x_{A,9} + F_{16} x_{A,14} + F_{19} x_{A,18} &= 30 \\ 0.333F_3 + F_7 x_{B,6} + F_{11} x_{B,9} + F_{16} x_{B,14} + F_{19} x_{B,18} &= 50 \\ 0.333F_3 + F_7 x_{C,6} + F_{11} x_{C,9} + F_{16} x_{C,14} + F_{19} x_{C,18} &= 30 \end{aligned}$$

(Compositions)

$$x_{A,i} + x_{B,i} + x_{C,i} = 1 \quad i = 5, 16, 9, 13, 14, 16$$

(Sharp Split)

$$x_{B,6} = x_{C,6} = x_{A,9} = x_{C,14} = x_{A,18} = x_{B,18} = 0.$$

By projecting on the compositions x_{A_i}, x_{B_i} and x_{C_i}, the primal and relaxed dual subproblems become linear. There are a total of 38 variables and 32 equality constraints. There are initially 20 *connected* variables (the flow rates). However, considering Figure 4.11, it is obvious that the recycle streams cannot both be simultaneously active. This leads to solving two independent problems, with $F_{10} = 0$ in the first case and $F_{15} = 0$ in the second case. In each case, the resulting problem has 9 *connected* variables. Application of the GOP algorithm to the problem identifies the global optimal solution (shown in Figure 4.12) in 17 iterations using the parallel configuration as a starting point. The total CPU time taken was 3.84 seconds on an HP730.

4.2.5 Concave and Indefinite Quadratic Problems

The GOP algorithm is primarily applicable to problems with quadratic functions in the objective and/or constraints. In this section, the results of applying the GOP and GOP/MILP algorithms to various problems of this type are discussed.

4.2.5.1 Problems from the literature. Eleven small-size concave quadratic problems from Phillips and Rosen (1988) have been solved using the GOP algorithm. The problems have the following form:

$$
\begin{aligned}
\min_{\boldsymbol{x},\boldsymbol{y}\in\Omega} \quad & \psi(\boldsymbol{x},\boldsymbol{y}) = \theta_1\varphi(\boldsymbol{x}) + \theta_2\boldsymbol{d}^T\boldsymbol{y} \\
s.t. \quad \varphi &= 0.5\sum_{i=1}^{n}\lambda_i(x_i - \overline{\omega_i})^2, \\
\Omega &= \{(x,y) : \boldsymbol{A}_1\boldsymbol{x} + \boldsymbol{A}_2\boldsymbol{y} \le \boldsymbol{b}, \boldsymbol{x} \ge 0, \boldsymbol{y} \ge 0\}, \\
\boldsymbol{x},\ \lambda,\ \overline{\omega} &\in \Re^n, \quad \boldsymbol{y},\ \boldsymbol{d} \in \Re^k \\
\boldsymbol{A}_1 &\in \Re^{m\times n}, \quad \boldsymbol{A}_2 \in \Re^{m\times k} \\
\theta_1,\ \theta_2 &\in \Re.
\end{aligned}
\tag{4.6}
$$

Here, m is the number of linear constraints, n is the number of concave variables ($\boldsymbol{x}$), and k is the number of linear variables ($\boldsymbol{y}$). The parameters θ_1 and θ_2 are -1 and 1 respectively, and the relative tolerance for convergence between the upper and lower bounds (ϵ) is 0.001.

The results of the application of the GOP algorithm to these problems are given in Table 4.5. The CPU times for the GOP algorithm and the Phillips and Rosen algorithm (denoted by P&R) are given in seconds. It should be noted that the P&R algorithm was run on a CRAY2. As can be seen, the algorithm solves problems of this size very fast, taking about 5 iterations to identify and converge to the global optimal solution.

Results from application of the GOP algorithm to another set of concave and indefinite quadratic test problems taken from Floudas and Pardalos (1990)

Problem	*Problem Size*			*GOP Algorithm*		P&R
	M	N	K	Iterations	CPU (HP730)	CPU (CRAY2)
1	5	2	0	3	0.09	0.026
2	5	6	0	2	0.07	0.022
3	5	6	0	2	0.06	0.020
4	5	6	0	2	0.03	0.026
5	4	2	0	4	0.12	0.017
6	4	3	0	4	0.11	0.015
7	4	3	0	4	0.14	0.014
8	10	3	0	17	0.50	0.022
9	10	3	0	8	0.20	0.020
10	4	4	0	3	0.18	0.029
11	9	2	1	3	0.08	0.023

Table 4.5. Test Problems from Phillips and Rosen (1988) ($\epsilon = 0.001$)

Problem	*Problem Size*			*GOP*		*Sherali & Tuncbilek*	
Name	N_X	N_Y	N_C	Iterations	CPU	Iterations	CPU
CQP1	10	10	11	27	0.68	32	0.15
CQP3	20	20	10	11	10.84	3	3.29
CQP4	20	20	10	4	3.57	1	2.61
CQP5	20	20	10	11	10.91	1	2.55
CQP6	20	20	10	5	5.07	1	2.61
CQP7	20	20	10	229	177.04	11	15.94
IQP1	20	20	10	3	0.65	3	2.73

Table 4.6. Quadratic Problems from Sherali and Tuncbilek (1994).

are given in table 4.6. These problems have also been solved recently by Sherali and Tuncbilek (1994) whose results are listed in the same table.

Run	Problem size			Iterations	CPU (sec)	
	m	n	k		GOP	GOP/MILP
CLR1	50	50	50	2.3	0.510	0.317
CLR2	50	50	100	3.0	5.736	2.254
CLR3	50	50	200	4.33	27.620	8.293
CLR4	50	50	300	5.0	—	8.977
CLR5	50	100	50	3.5	32.07	5.665
CLR6	50	100	150	6.8	—	38.892
CLR7	100	100	100	2.2	3.485	31.147
CLR8	100	200	100	3.8	—	100.370
CLR9	100	250	100	3.6	—	267.124

Table 4.7. Concave Quadratic Problems from Phillips and Rosen (1988), $\epsilon = 0.01$

Run	Problem size			Iterations	CPU (sec)	
	m	n	k		GOP	GOP/MILP
CLR1	50	50	50	2.0	0.120	0.116
CLR2	50	50	100	2.0	0.145	0.141
CLR3	50	50	200	2.2	6.047	1.574
CLR4	50	50	500	3.0	—	14.125
CLR5	50	100	100	2.0	0.217	1.373
CLR6	50	100	200	2.0	0.360	11.982
CLR7	100	100	100	2.0	0.305	0.306
CLR8	100	100	200	2.0	0.374	0.369
CLR9	100	100	200	2.0	0.374	0.369
CLR10	100	100	500	3.0	—	80.028
CLR11	100	150	400	1.7	—	182.208

Table 4.8. Concave Quadratic Problems from Phillips and Rosen (1988), $\epsilon = 0.1$

Run	Problem size			$\epsilon = 0.1$		$\epsilon = 0.01$	
	m	n	k	Iter	CPU	Iter	CPU
ILR1	25	25	25	2.0	0.232	2.200	0.312
ILR2	25	25	50	2.0	0.416	2.600	0.606
ILR3	25	25	100	2.2	1.522	3.000	2.030
ILR4	25	50	100	4.0	13.19	11.50	37.56
ILR5	50	50	50	2.0	0.864	2.400	1.504
ILR6	50	50	100	2.0	1.264	2.800	3.018
ILR7	25	75	100	3.0	68.86	30.00	294.3
ILR8	50	75	100	2.0	1.564	3.600	9.724
ILR9	75	75	100	2.0	2.120	2.800	6.304
ILR10	25	75	150	4.0	115.80	—-	—-
ILR11	50	75	150	2.2	9.5380	—-	—-
ILR12	75	75	150	2.0	2.9560	—-	—-
ILR13	25	100	50	3.6	23.21	23.50	118.6
ILR14	50	100	50	2.2	2.130	3.800	6.510
ILR15	75	100	50	2.2	3.544	2.800	5.244

Table 4.9. Indefinite Quadratic Problems from Phillips and Rosen (1988), $\epsilon = 0.1$ and 0.01

4.2.5.2 Randomly Generated Quadratic Problems. This section describes the application of the GOP and GOP/MILP algorithms to randomly generated problems of the form (4.6). Such problems have earlier been studied by Phillips and Rosen (1988), and we generated the data for the constants $\lambda, \overline{\omega}, \boldsymbol{d}, \boldsymbol{A}_1, \boldsymbol{A}_2$ and $\boldsymbol{b}$ as they have used. The parameters θ_1 and θ_2 have been set to values of -0.001 and 0.1 respectively.

Depending on the values of λ_i, the problems generated are either concave quadratic or indefinite quadratic problems. For the case of indefinite quadratic problems, roughly as many positive λ_i as negative λ_i are generated. For each problem size, 5-10 different problems (using various seeds) have been generated and solved.

Tables 4.7 and 4.8 present the results for concave quadratic problems using tolerances of 0.01 and 0.1 respectively, while Table 4.9 presents the results for

indefinite quadratic problems using tolerances of 0.01 and 0.1 with the GOP algorithm. In all the cases, it can be seen that the algorithm generally requires very few iterations for the upper and lower bounds to be within 10% of the optimal solution; generally, the convergence to within 1% is achived in a few more iterations. Moreover, certain trends are noticeable in all cases. For example, as the number of constraints (m) grows, the problems generally become easier to solve. Conversely, as the size of the linear variables (k) increases, the algorithm requires more time for the solution of the dual problems, leading to larger overall CPU times.

4.3 SUMMARY

In this chapter, we have discussed the implementation of various versions of the GOP algorithm. The new framework as a branch and bound global optimization algorithm permits a simplified implementation. The resulting package *cGOP* has been applied to a large number of engineering design and control problems, as well as indefinite and concave quadratic problems. The implementation permits very efficient solutions of problems of medium size, as evidenced by the reported computational results.

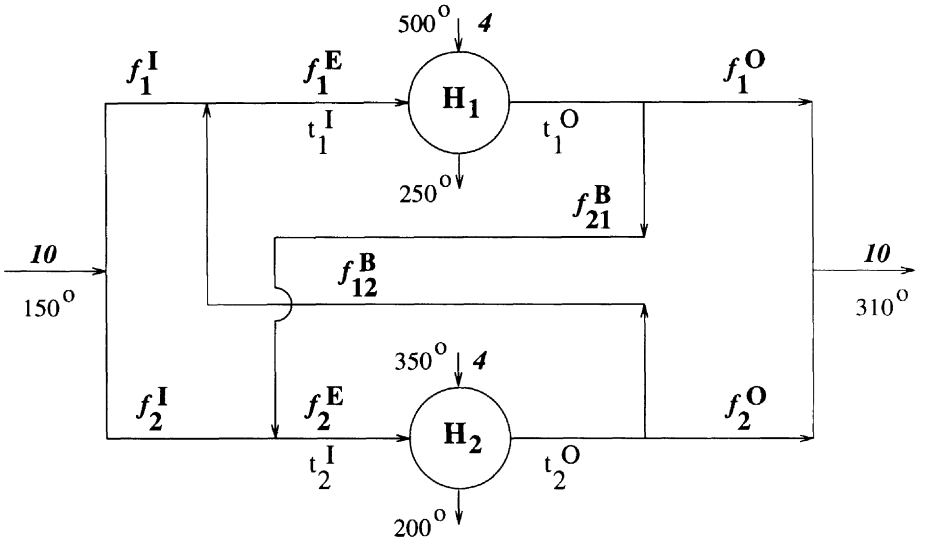


Figure 4.5. Heat Exchanger Network Superstructure For Example 4.2.1

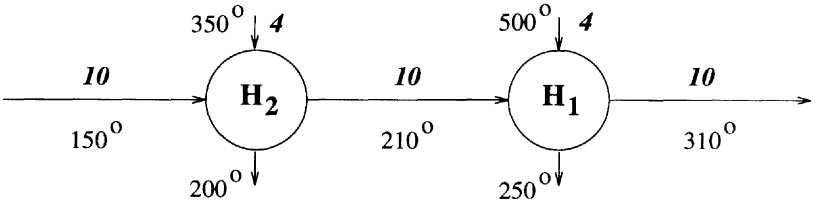


Figure 4.6. Optimal Configuration For Example 4.2.1

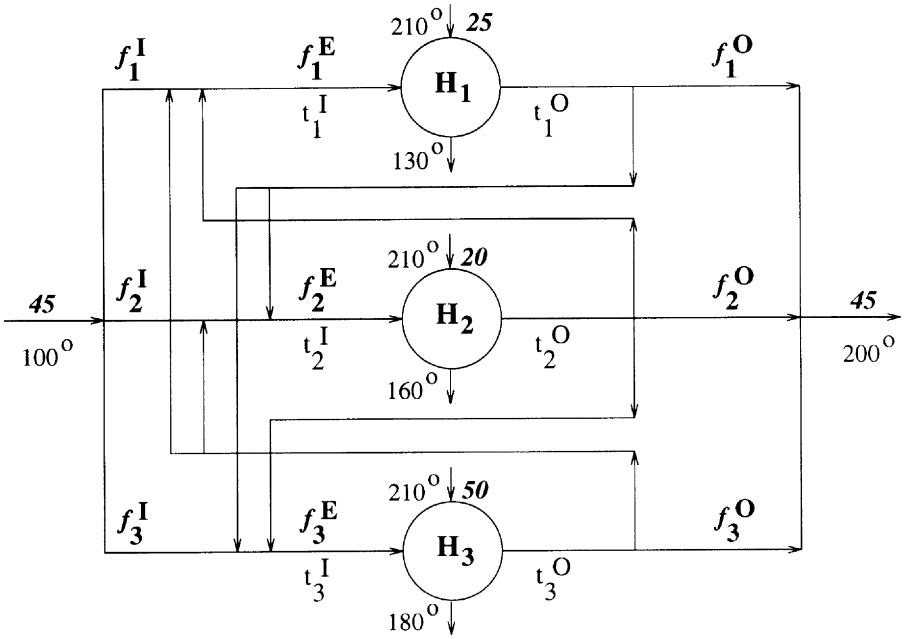


Figure 4.7. Heat Exchanger Network Superstructure For Example 4.2.2

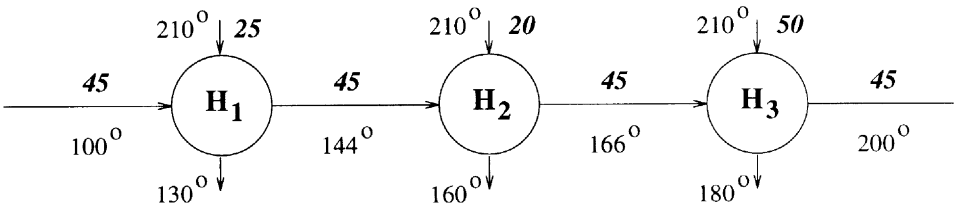


Figure 4.8. Optimal Configuration For Example 4.2.2

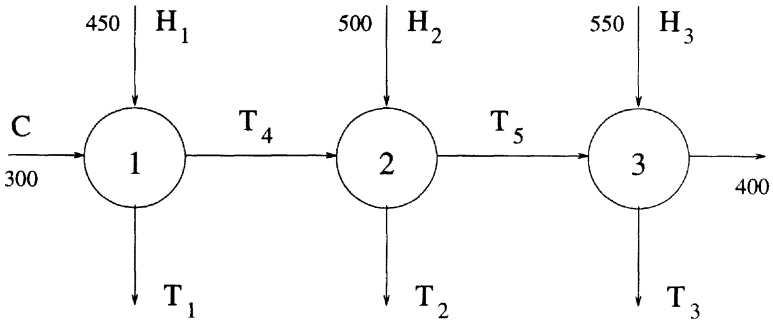


Figure 4.9. Heat Exchanger Example From Quesada and Grossmann (1993)

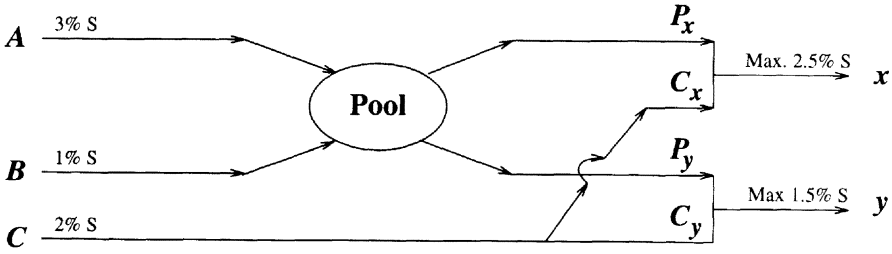


Figure 4.10. The Haverly Pooling Problem

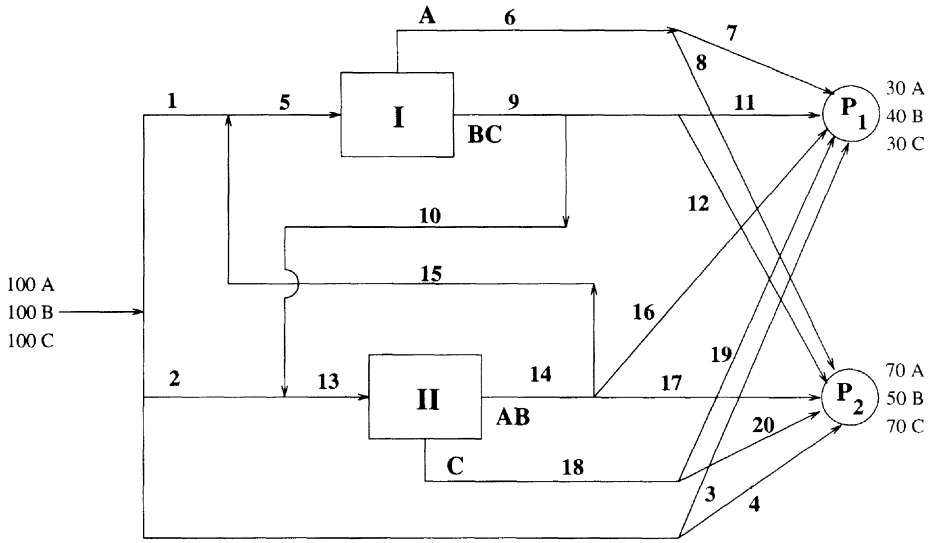


Figure 4.11. Superstructure for Example ??

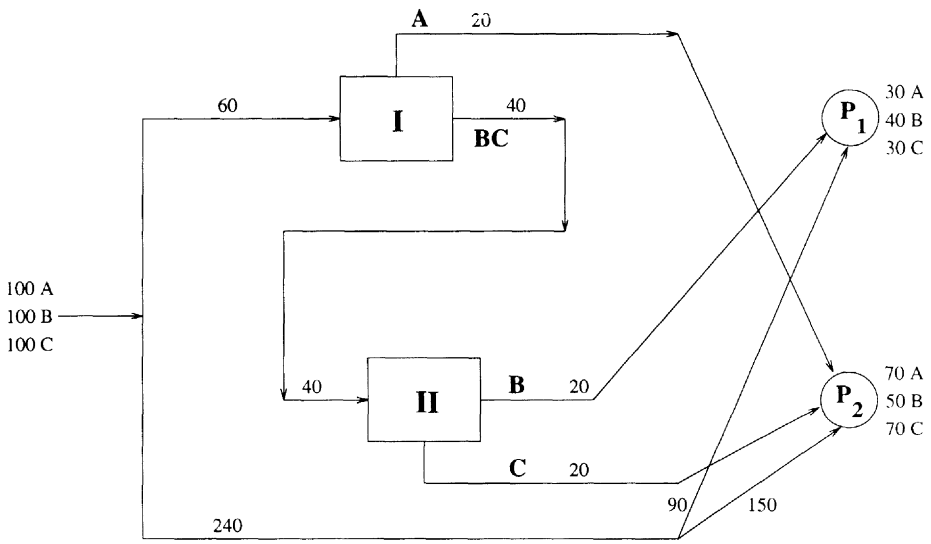


Figure 4.12. Optimal Configuration For Example ??

5 THE GOP APPROACH IN BILEVEL LINEAR AND QUADRATIC PROBLEMS

This chapter presents the primal-relaxed dual decomposition based global optimization approach GOP for bilevel linear and quadratic programming problems as developed by Visweswaran *et al.* (1996). In section 5.1, the reader is briefly introduced to the bilevel programming literature work. In section 5.2, the bilevel linear problem is formulated followed by the theoretical analysis based on the principles of the GOP algorithm. Section 5.3 presents the modified-GOP approach along with computational studies for bilevel linear programming problems. Finally, section 5.4 presents the theoretical and computational studies for bilevel quadratic programming models based on the GOP principles.

5.1 INTRODUCTION

Bilevel programming refers to optimization problems in which the constraint region is implicitly determined by another optimization problem, as follows:

$$\begin{aligned}
&\min_{\boldsymbol{x}} \quad F(\boldsymbol{x},\boldsymbol{y}) \\
&s.t. \\
&\boldsymbol{G}(\boldsymbol{x},\boldsymbol{y}) \leq \boldsymbol{0} \\
&\boldsymbol{y} \in \left\{ \begin{array}{rl} \min\limits_{\boldsymbol{y}} & f(\boldsymbol{x},\boldsymbol{y}) \\ s.t. & \boldsymbol{g}(\boldsymbol{x},\boldsymbol{y}) \leq \boldsymbol{0} \\ \boldsymbol{x} \in X, & \boldsymbol{y} \in Y \end{array} \right\}
\end{aligned} \tag{5.1}$$

C. A. Floudas, *Deterministic Global Optimization*
© Springer Science+Business Media Dordrecht 2000

where $\boldsymbol{G}(\boldsymbol{x}, \boldsymbol{y})$ is the vector valued function $X \times Y \to R^p$, $\boldsymbol{g}(\boldsymbol{x}, \boldsymbol{y})$ is the vector valued function $X \times Y \to R^m$, and X and Y are compact convex sets.

Problem (5.1) can be interpreted in the following way. At the higher level the decision maker (leader) chooses first a vector $\boldsymbol{x} \in X$ to minimize the objective function F; then in light of this decision the lower level decision maker (follower) selects the decision vector $\boldsymbol{y} \in Y$ that minimizes the objective function f.

Applications of bilevel programming are diverse and include (i) design optimization problems of chemical plants where regions of different models should be examined (e.g., in equilibrium calculations where the different regions correspond to different number and type of phases), (ii) long-range planning problems followed by short-term scheduling in chemical and other industries, (iii) hierarchical decision making policy problems in mixed economies, where policy makers at the top level influence the decisions of private individuals and companies, (iv) energy consumption of private companies, which is affected by imported resources controlled by government policy, (v) parameter estimation problems coupled with prediction of thermodynamic properties, and (vi) location-allocation problems that arise in the optimal placement of platforms and wells.

Problem (5.1) has received a lot of attention, especially for the linear case. There are two major classes of approaches for bilevel linear problems: (i) enumeration methods, and (ii) reformulation methods.

(i) Enumeration techniques exploit the fact that an optimal solution to the bilevel problem is a basic feasible solution of the linear constraints involved at the lower and upper level and consequently must occur at an extreme point of the feasible set (e.g., the enumeration method by Candler and Townsley (1982); "Kth Best algorithm" by Bialas and Karwan (1984); branch and bound algorithm by Bard and Moore (1990)).

(ii) Reformulation techniques based on the transformation of the original problem to a single optimization problem by employing the optimality KKT conditions of the lower level problem. For the solution of the resulting formulation the following algorithms have been developed: B&B techniques (Bard and Falk (1982)); mixed integer programming techniques (Fortuny-Amat and McCarl (1981)); parametric complementarity pivoting (Judice and Faustino (1992)); local optimization approaches for nonlinear programming such as penalty and barrier function methods (Anandalingam and White (1990), White and Anandalingam (1993)) and global optimization techniques based on the reformulation of the complementarity slackness constraint to a separable quadratic reverse convex inequality constraint (Al-Khayyal *et al.* (1992)) or the restatement of the original problem as a reverse convex program (Tuy *et al.* (1993a), Tuy *et al.* (1993b)).

For bilevel nonlinear problems, Bard (1983), (1984) developed an one dimensional search algorithm that yields a locally optimal solution. Clark and Westerberg (1988), Ben-Ayed and Blair (1989) and Haurie *et al.* (1990) showed that the optimality conditions used by Bard (1983), (1984) are not correct.

Penalty function methods were used by Aiyoshi and Shimizu (1981), which do not guarantee the global optimal solution because of the non-convex nature of the problem.

5.2 BILEVEL LINEAR PROGRAMMING PROBLEM

If all functions are linear, problem (5.1) gives rise to the following bilevel linear programming formulation:

$$\begin{aligned}
&\min_{\boldsymbol{x}} \quad F(\boldsymbol{x},\boldsymbol{y}) = \boldsymbol{c}_1^T\boldsymbol{x} + \boldsymbol{d}_1^T\boldsymbol{y} \\
&s.t. \\
&\boldsymbol{G}(\boldsymbol{x},\boldsymbol{y}) \leq \boldsymbol{0} \\
&\boldsymbol{y} \in \left\{ \begin{array}{rl} \min\limits_{\boldsymbol{y}} & f(\boldsymbol{x},\boldsymbol{y}) = \boldsymbol{c}_2^T\boldsymbol{x} + \boldsymbol{d}_2^T\boldsymbol{y} \\ s.t. & \boldsymbol{g}(\boldsymbol{x},\boldsymbol{y}) = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{y} - \boldsymbol{b} \quad \leq \quad \boldsymbol{0} \\ & \boldsymbol{x} \geq \boldsymbol{0} \end{array} \right\}.
\end{aligned} \tag{5.2}$$

For the sake of simplicity, the constraints $\boldsymbol{G}(\boldsymbol{x},\boldsymbol{y})$ will be ignored in the sequel. It is easy to show that the results obtained below hold in the presence of general convex constraints at the outer level. It should also be noted that any bounds on $\boldsymbol{y}$ are assumed to be incorporated into the inner level inequality constraints. For the rest of this chapter the following terminology will be used:

Follower's Feasible region :

$$S(\boldsymbol{x}) = \{\boldsymbol{y} \mid \boldsymbol{g}(\boldsymbol{x},\boldsymbol{y}) \leq \boldsymbol{0}\}.$$

Follower's Rational Reaction Set :

$$RR(\boldsymbol{x}) = \{\boldsymbol{y} \in argmin f(\boldsymbol{x},\boldsymbol{y}) \mid \boldsymbol{y} \in S(\boldsymbol{x})\}.$$

Inducible Region :

$$IR = \{(\boldsymbol{x},\boldsymbol{y}) \mid \boldsymbol{x} \geq \boldsymbol{0}, \boldsymbol{y} \in RR(\boldsymbol{x})\}.$$

The solution of bilevel linear programming problems involves a number of interesting features:

- even for the linear case where the follower's feasible region S is convex, the inducible region (IR) where the leader's objective should be minimized is a non-convex region;
- Hansen *et al.* (1992), proved that the bilevel linear programming problem is strongly **NP**-hard; and
- The presence of dual degeneracy in the follower's problem, while not affecting the value of follower's objective function, can have an impact on the leader's objective. In this case the follower's choice among the multiple optima will

be based on his willingness to cooperate with the leader. Two extreme cases concern the follower's acceptance of the leader's preferences regarding $\boldsymbol{y}$ (the tie cooperative case) and the follower's adopting the opposite of the leader's preferences (the tie non-cooperative case). The approach presented in this chapter can handle both cases, but can be expected to be more efficient for the tie cooperative case.

5.2.1 Equivalent Formulation

Rather than working with problem (5.2) in its hierarchical form the analysis begins by converting it into a single mathematical program. This can be achieved by replacing the follower's optimization problem with the necessary and sufficient KKT optimality conditions. This results in the following problem:

$$\begin{aligned}
&\min_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{u}} \quad \boldsymbol{c}_1^T\boldsymbol{x} + \boldsymbol{d}_1^T\boldsymbol{y} \\
&s.t. \;\; \boldsymbol{d}_2^T + \boldsymbol{u}^T\boldsymbol{B} = \boldsymbol{0} \\
&u_i(\boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{y} - \boldsymbol{b})_i = 0, i = 1,..,m \\
&\boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{y} \leq \boldsymbol{b} \\
&\boldsymbol{x} \geq \boldsymbol{0}, \boldsymbol{y} \geq \boldsymbol{0}, u_i \geq 0, i = 1,..,m
\end{aligned} \tag{5.3}$$

where u_i is the Lagrange multiplier of the ith follower's constraint $(\boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{y} - \boldsymbol{b})_i$, $i = 1,..,m$. Note that the optimality conditions assume the existence of a stable point for the inner optimization problem, and therefore assume the satisfaction of an appropriate first order constraint qualification. Problem (5.3) is a single nonlinear optimization problem, albeit non-convex due to the presence of bilinear terms in the complementarity conditions. In this chapter, we study further this bilinear structure of Problem (5.3) based on the principles of the GOP global optimization approach presented in Chapter 3.

5.2.2 Mathematical Properties

Consider the following partition of the variables $Y = \boldsymbol{u}, X = (\boldsymbol{x}, \boldsymbol{y})$ which satisfies Conditions (A) of the GOP algorithm presented in chapter 3 (Floudas and Visweswaran, 1990, 1993). For fixed $Y = Y^k$, the primal problem can be written as:

$$\begin{aligned}
&\min_{\boldsymbol{x},\boldsymbol{y}} \quad \boldsymbol{c}_1^T\boldsymbol{x} + \boldsymbol{d}_1^T\boldsymbol{y} \\
&s.t. \;\; \boldsymbol{d}_2^T + \boldsymbol{u}^T\boldsymbol{B} = \boldsymbol{0} \\
&Y_i^k(\boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{y} - \boldsymbol{b})_i = 0, i = 1,..,m \\
&\boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{y} \leq \boldsymbol{b} \\
&\boldsymbol{x} \geq \boldsymbol{0}, \boldsymbol{y} \geq \boldsymbol{0}.
\end{aligned} \tag{5.4}$$

Note that the KKT gradient conditions in problem (5.3), which are in the variables $\boldsymbol{u}$, can be used directly in the dual problem. The solution to this

primal problem, if feasible, yields the multipliers λ^k and μ^k for the equality and inequality constraints. Note that when $u_i^k = 0$, the corresponding constraint drops out from the set of equality constraints, and there will be no multiplier for that constraint, implying that $\lambda_i^k = 0$ for this case. Conversely, when $u_i^k > 0$, the corresponding constraint is active, and therefore the value of μ_i^k is zero.

The relaxed dual problem takes the form:

$$\begin{aligned} \min_{\mu_B, \boldsymbol{u}} \quad & \mu_B \\ \text{s.t.} \quad & \mu_B \geq \min_{\boldsymbol{x}, \boldsymbol{y}} L(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{u}, \mu^k, \lambda^k) \end{aligned}$$

where $L(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{u}, \mu^k, \lambda^k)$ is the Lagrange function of the primal problem and is given by

$$L(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{u}, \mu^k, \lambda^k) = \boldsymbol{c}_1^T \boldsymbol{x} + \boldsymbol{d}_1^T \boldsymbol{y} + \sum_{i=1}^{m} (\mu_i^k + \lambda_i^k u_i)(\boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{y} - \boldsymbol{b})_i \quad (5.5)$$

where $(\boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{y} - \boldsymbol{b})_i$ refers to the ith inner constraint. Separating the terms in $\boldsymbol{x}$ and $\boldsymbol{y}$, equation (5.5) can be rewritten as:

$$\begin{aligned} L(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{u}, \mu^k, \lambda^k) = \; & [\boldsymbol{c}_1 + \sum_{i=1}^{m} (\mu_i^k + \lambda_i^k u_i) \boldsymbol{A}_i]^T \boldsymbol{x} + \\ & [\boldsymbol{d}_1 + \sum_{i=1}^{m} (\mu_i^k + \lambda_i^k u_i) \boldsymbol{B}_i]^T \boldsymbol{y} - \sum_{i=1}^{m} (\mu_i^k + \lambda_i^k u_i) b_i \end{aligned} \quad (5.6)$$

where $\boldsymbol{A}_i$ and $\boldsymbol{B}_i$ are the ith rows of $\boldsymbol{A}$ and $\boldsymbol{B}$ respectively. The KKT gradient conditions for fixed Lagrange multipliers $Y = Y^k = \boldsymbol{u}^k$, can be written as:

$$\nabla_{\boldsymbol{x}} L(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{u}^k, \mu^k, \lambda^k) = \boldsymbol{c}_1 + \sum_{i=1}^{m} \mu_i^k \boldsymbol{A}_i + \sum_{i=1}^{m} \lambda_i^k u_i^k \boldsymbol{A}_i = \boldsymbol{0} \quad (5.7)$$

$$\nabla_{\boldsymbol{y}} L(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{u}^k, \mu^k, \lambda^k) = \boldsymbol{d}_1 + \sum_{i=1}^{m} \mu_i^k \boldsymbol{B}_i + \sum_{i=1}^{m} \lambda_i^k u_i^k \boldsymbol{B}_i = \boldsymbol{0}. \quad (5.8)$$

Using the KKT gradient conditions in the Lagrange function results in :

$$L(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{u}, \mu^k, \lambda^k) = \sum_{i=1}^{m} \left[\lambda_i^k (u_i - u_i^k) S_i - (\mu_i^k + \lambda_i^k u_i^k) b_i \right] \quad (5.9)$$

where $\boldsymbol{S} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{y} - \boldsymbol{b}$ are slacks introduced for convenience. The advantage of this expression for the Lagrange function is that the following property can be established which effectively allows the relaxed dual to be replaced by a sequence of subproblems corresponding to combinations of lower and upper bounds of the constraints.

Property 5.2.1 *Suppose that the minimum value of the Lagrange function* L^*, $L^*(\bar{\boldsymbol{x}}, \bar{\boldsymbol{y}}, \boldsymbol{u}, \mu^k, \lambda^k) = \min_{\boldsymbol{x},\boldsymbol{y}} L(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{u}, \mu^k, \lambda^k)$ *occurs at* $(\bar{\boldsymbol{x}}, \bar{\boldsymbol{y}})$*; then,*

$$L^*(\bar{\boldsymbol{x}}, \bar{\boldsymbol{y}}, \boldsymbol{u}, \mu^k, \lambda^k) \geq \min_{B_j \in CB} \left\{ \begin{array}{ll} \sum_{i=1}^{m} [\lambda_i^k (u_i - u_i^k) S_i^{B_j} - (\mu_i^k + \lambda_i^k u_i^k) b_i] & \\ \lambda_i^k (u_i - u_i^k) \leq 0, & \forall S_i^{B_j} = S_i^L \\ \lambda_i^k (u_i - u_i^k) \geq 0, & \forall S_i^{B_j} = S_i^U \end{array} \right\}$$

where $\boldsymbol{S} = (\boldsymbol{Ax} + \boldsymbol{By} - \boldsymbol{b})$ *are slacks* ($\boldsymbol{S} \geq \boldsymbol{0}$) *introduced for ease in the presentation;* S_i^L, S_i^U *are the lower and upper bounds on the constraints* $(\boldsymbol{Ax} + \boldsymbol{By} - \boldsymbol{b})_i$*, respectively;* B_j *corresponds to a combination of lower/upper bounds of constraints;* S^{B_j} *is the vector of lower/upper bounds of the constraints corresponding to the bound combination* B_j*; and* CB *is the set of all bound combinations.*

Proof. Consider the ith component of the Lagrange function, corresponding to the ith inner constraint. The minimum of this constraint lies at a bound of $(\boldsymbol{Ax} + \boldsymbol{By} - \boldsymbol{b})_i$, the actual bound being determined by the sign of $\lambda_i^k (u_i - u_i^k)$. The following two cases can be distinguished:

(a) If $u_i \geq u_i^k$ and $\lambda_i^k \leq 0$, or if $u_i \leq u_i^k$ and $\lambda_i^k \geq 0$,

$$\min_{\boldsymbol{x},\boldsymbol{y}} L(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{u}, \mu^k, \lambda^k) \geq \sum_{i=1}^{m} [\lambda_i^k (u_i - u_i^k) S_i^U - (\mu_i^k + \lambda_i^k u_i^k) b_i].$$

(b) If $u_i \leq u_i^k$ and $\lambda_i^k \geq 0$, or if $u_i \geq u_i^k$ and $\lambda_i^k \leq 0$,

$$\min_{\boldsymbol{x},\boldsymbol{y}} L(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{u}, \mu^k, \lambda^k) \geq \sum_{i=1}^{m} [\lambda_i^k (u_i - u_i^k) S_i^L - (\mu_i^k + \lambda_i^k u_i^k) b_i].$$

Then, it follows that there exists a combination of bounds B_j of the constraints such that:

$$\min_{\boldsymbol{x},\boldsymbol{y}} L(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{u}, \mu^k, \lambda^k) \geq \sum_{i=1}^{m} [\lambda_i^k (u_i - u_i^k) S_i^{B_j} - (\mu_i^k + \lambda_i^k u_i^k) b_i].$$

Remark 5.2.1 The above property preserves the important feature of the GOP algorithm that the solution of the relaxed dual problem can be equivalently substituted by a series of optimization subproblems corresponding to different partitions of the Y-space.

Remark 5.2.2 Note that the Lagrange function is essentially expressed in terms of the follower's constraints. This implies that from a computational point of view, the complexity of the relaxed dual problem is determined by the number of active inner problem constraints (i.e., those constraints for which $\lambda_i^k \neq 0$). This can be of great significance in problems with large number of variables but few constraints.

5.2.3 *Introduction of 0-1 variables*

It is clear that each combination of the u variables corresponds to a vertex of the follower's feasible region. However, different combinations with the same set of nonzero u_i correspond to the same vertex. It is desirable to avoid such nonzero combinations from being generated more than once. This can be ensured by the introduction of binary variables, as shown below.

Consider the set of binary variables a_i, $i = 1, \ldots, m$, associated with each one of the follower's constraints as follows:

$$a_i = \begin{cases} 1, \text{if constraint } (\boldsymbol{Ax} + \boldsymbol{By} - \boldsymbol{b})_i \text{ is active} \\ 0, \text{otherwise} \end{cases}$$

The following set of *big-M* constraints are also introduced to establish one-to-one correspondence between the multiplier u_i of constraint i and the corresponding 0-1 variable a_i:

$$(1/M)a_i \leq u_i \leq Ma_i. \tag{5.10}$$

This constraint implies that if $a_i = 0 \Rightarrow 0 \leq u_i \leq 0 \Rightarrow u_i = 0$, i.e., the multiplier is also zero, forcing the corresponding constraint to be tight, whereas if $a_i = 1 \Rightarrow (1/M) \leq u_i \leq M$, the associated multiplier has nonzero value implying an inactive constraint.

The incorporation of the aforementioned constraints along with the 0-1 variables a_i into problem (5.3) results in:

$$\begin{aligned} & \min_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{u}} \quad \boldsymbol{c}_1^T\boldsymbol{x} + \boldsymbol{d}_1^T\boldsymbol{y} \\ \text{s.t.} \quad & \boldsymbol{d}_2^T + \boldsymbol{u}^T\boldsymbol{B} = \boldsymbol{0} \\ & a_i(\boldsymbol{Ax} + \boldsymbol{By} - \boldsymbol{b})_i = 0, \quad i = 1, .., m \\ & u_i \leq Ma_i, \quad i = 1, .., m \\ & a_i \leq Mu_i, \quad i = 1, .., m \\ & \boldsymbol{Ax} + \boldsymbol{By} \leq \boldsymbol{b} \\ & \boldsymbol{x} \geq \boldsymbol{0}, \ \boldsymbol{y} \geq \boldsymbol{0}, \ \boldsymbol{u} \geq \boldsymbol{0}, \ a_i = \{0 - 1\}. \end{aligned}$$

By augmenting the Y-vector to include the 0-1 variables, the following primal problem can be derived for $Y = Y^k = (\boldsymbol{u}^k, \boldsymbol{a}^k)$:

$$\begin{aligned} & \min_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{u}} \quad \boldsymbol{c}_1^T\boldsymbol{x} + \boldsymbol{d}_1^T\boldsymbol{y} \\ \text{s.t.} \quad & a_i^k(\boldsymbol{Ax} + \boldsymbol{By} - \boldsymbol{b})_i = 0, \quad i = 1, .., m \\ & \boldsymbol{Ax} + \boldsymbol{By} \leq \boldsymbol{b} \\ & \boldsymbol{x} \geq \boldsymbol{0}, \ \boldsymbol{y} \geq \boldsymbol{0} \end{aligned}$$

with a corresponding relaxed dual problem of the following form:

$$\begin{aligned}
&\min_{\mu_B, \boldsymbol{u}, \boldsymbol{a}} \quad \mu_B \\
\text{s.t.} \quad &\mu_B \geq \min_{\boldsymbol{x}, \boldsymbol{y}} \quad L(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{a}, \mu^k, \lambda^k) \\
&L(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{a}, \mu^k, \lambda^k) = \boldsymbol{c}_1^T \boldsymbol{x} + \boldsymbol{d}_1^T \boldsymbol{y} + \sum_{i=1}^{m} (\mu_i^k + \lambda_i^k a_i)(\boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{y} - \boldsymbol{b})_i \\
&u_i \leq M a_i, \quad i = 1, .., m \\
&a_i \leq M u_i, \quad i = 1, .., m.
\end{aligned}$$

Using the KKT optimality conditions for the primal problem the Lagrange function can be written as

$$L(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{a}, \mu^k, \lambda^k) = \sum_{i=1}^{m} [\lambda_i^k (a_i - a_i^k)(\boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{y} - \boldsymbol{b})_i - (\mu_i^k + \lambda_i^k a_i^k) b_i]. \quad (5.11)$$

Property 5.2.1 can then be recast as follows:

$$L^*(\bar{\boldsymbol{x}}, \bar{\boldsymbol{y}}, \boldsymbol{u}, \mu^k, \lambda^k) \geq \min_{B_j \in CB} \left\{ \begin{array}{ll} \sum_{i=1}^{m} [\lambda_i^k (a_i - a_i^k) S_i^{B_j} - (\mu_i^k + \lambda_i^k a_i^k) b_i] & \\ \lambda_i^k (a_i - a_i^k) \leq 0, & \forall S_i^{B_j} = S_i^L \\ \lambda_i^k (a_i - a_i^k) \geq 0, & \forall S_i^{B_j} = S_i^U \end{array} \right\}$$

Consider the ith term. It is clear that if $a_i^k = 0$, the corresponding constraint would have been absent from the primal problem, leading to $\lambda_i^k = 0$, so that this term would be absent from the summation. Therefore, only the case of $a_i^k = 1$ is important. Then, since a_i is always less than or equal to a_i^k, the minimum of $L(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{a}, \mu^k, \lambda^k)$ occurs at the lower (upper) bound of $(\boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{y} - \boldsymbol{b})_i$ if $\lambda_i^k \leq 0 (\lambda_i^k \geq 0)$. Therefore, it is sufficient to set each active constraint in the summation to the appropriate bound, and the following result is always true:

Only one relaxed dual problem is solved at every iteration regardless of the size of the problem.

Remark 5.2.3 Additional constraints (integer cuts) in the 0-1 variables, a_i, can be used together with the Lagrangian cut to improve the solution efficiency of the resulting MILP relaxed dual problem. In particular, as has been shown by Hansen et al. (1992), in any optimal solution of bilevel programming problem the active constraints of the follower's problem satisfy the following conditions:

$$\sum_{i}^{I_p(i)} a_i \geq 1, \text{ if } d_i > 0, \ i = 1, ...m$$

$$\sum_{i}^{I_n(i)} a_i \geq 1, \text{ if } d_i < 0, \ i = 1, ...m$$

where $I_p(i), I_n(i)$ are the sets of constraints in which variable y_i appears with positive and negative sign, respectively. Also, an active set strategy suggests

that:

$$\sum_{i=1}^{m} a_i \leq |\boldsymbol{y}|$$

where $|\boldsymbol{y}|$ is the cardinality of the follower's decision vector $\boldsymbol{y}$. It can be seen that these and other preprocessing steps can be done on the binary variables to eliminate certain combinations.

Remark 5.2.4 In cases where the primal problem is infeasible, the following relaxed primal problem is formulated and solved:

$$\begin{aligned} \min_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{s}_I,\boldsymbol{s}_A^+,\boldsymbol{s}_A^-} \quad & \theta = \sum_{i=1}^{m_A}(s_{Ai}^+ + s_{Ai}^-) + \sum_{i=1}^{m_I} s_{Ii} \\ \text{s.t.} \quad & \boldsymbol{a}^{kT}(\boldsymbol{A}_A\boldsymbol{x} + \boldsymbol{B}_A\boldsymbol{y} - \boldsymbol{b}_A) + \boldsymbol{s}_A^+ - \boldsymbol{s}_A^- = \boldsymbol{0} \\ & \boldsymbol{A}_I\boldsymbol{x} + \boldsymbol{B}_I\boldsymbol{y} - \boldsymbol{b}_I - \boldsymbol{s}_I \leq \boldsymbol{0} \\ & \boldsymbol{x}, \boldsymbol{y}, \boldsymbol{s}_A^+, \boldsymbol{s}_A^-, \boldsymbol{s}_I \geq \boldsymbol{0} \end{aligned}$$

where m_A, m_I are the number of active and inactive constraints at the current iteration, respectively; $\boldsymbol{s}_I, \boldsymbol{s}_A^+, \boldsymbol{s}_A^-$ are vectors of slacks variables that are introduced in order to minimize the sum of infeasibilities. The Lagrange function takes the form:

$$\begin{aligned} L(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{s}_I, \boldsymbol{s}_A^+, \boldsymbol{s}_A^-) = \theta + \sum_{i=1}^{m} \{ & \lambda_i^k[a_i(\boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{y} - \boldsymbol{b}) + s_{Ai}^+ - s_{Ai}^-] \\ & + \mu_i^k[\boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{y} - \boldsymbol{b} - s_{Ii}]_i \} \end{aligned} \tag{5.12}$$

which can be transformed by using the optimality conditions to :

$$L(\boldsymbol{x}, \boldsymbol{y}) = \sum_{i=1}^{m} \lambda_i^k(a_i - a_i^k)(\boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{y} - \boldsymbol{b})_i + \bar{\theta} \tag{5.13}$$

where $\bar{\theta}$ is the optimal solution of the feasibility problem. Based on this, the following feasibility cut can be introduced in the relaxed dual problem :

$$\sum_{i=1}^{m} \lambda_i^k(a_i - a_i^k)(\boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{y} - \boldsymbol{b})_i + \bar{\theta} \leq 0. \tag{5.14}$$

5.3 MODIFIED GOP ALGORITHM

Based on the above analysis, a modified GOP algorithm for global optimization of bilevel linear programming programs is now described in the following steps:

Step 0: *Initialization of Parameters.*

Define the storage parameters $\mu_B^{stor}(K^{max}), \boldsymbol{y}^{stor}(K^{max})$ and $\boldsymbol{y}^k(K^{max})$ over

the maximum expected number of iterations K^{max}. Define P^{UBD}, R^{LBD} as the upper and lower bounds obtained from the primal and relaxed dual problems, respectively. Set $\mu_B^{stor}(K^{max}) = U, P^{UBD} = U, R^{LBD} = L$, where U and L are very large positive and negative numbers, respectively. Select a starting point Y^1, set the iteration counter equal to 1, and select a tolerance for convergence ϵ. Find lower and upper bounds on the inner constraints by solving the following problems:

$$\left.\begin{array}{ll} \min\limits_{\boldsymbol{x},\boldsymbol{y}} & \pm(\boldsymbol{Ax}+\boldsymbol{By}-\boldsymbol{b})_i \\ \text{s.t.} & (\boldsymbol{Ax}+\boldsymbol{By}-\boldsymbol{b})_j \leq 0 \; j=1,..,m, \; j \neq i \\ & \boldsymbol{x},\boldsymbol{y} \geq \boldsymbol{0} \end{array}\right\} \; (PB^i) \; i=1,...,m.$$

Step 1: *Primal Problem.*

Store the value Y^k. Solve the primal problem and store the Lagrange multipliers λ_A^k and update the upper bound so that $P^{UBD} = \min\{P^{UBD}, Z^k\}$, where Z^k is the current primal objective. If the primal is infeasible a relaxed primal is solved and the Lagrange multipliers are stored.

Step 2: *Relaxed-Dual Problem.*

Formulate the Lagrange function corresponding to the current primal problem and add this as a constraint to the relaxed dual problem. For iterations when the primal problem is infeasible use a feasibility cut in the constraints corresponding to that iteration. Solve the resulting relaxed dual problem and store the solution $\mu_B^{stor}(k), \boldsymbol{y}^{stor}(k)$.

Step 3: *Selecting a new lower bound R^{LBD}.*

From the stored values μ_B^{stor} select the minimum μ_B^{min} and set $R^{LBD} = \mu_B^{min}$, $Y^{k+1} = Y^{min}$ the corresponding stored value of Y. Delete μ_B^{min}, Y^{min} from the stored set.

Step 4: *Check for convergence.*

Check if $R^{LBD} \geq P^{UBD} - \epsilon$. If yes, stop. Else set $k = k+1$ and return to Step 1.

5.3.1 Illustrative Example

Consider the following two-level linear program (Bard, 1983):

$$\begin{array}{ll} \min\limits_{x} & F(x,y) = x+y \\ \text{s.t.} & -x \leq 0 \\ & \quad \min\limits_{y} \quad f(x,y) = -5x-y \\ & \quad \text{s.t.} \quad -x-0.5y \leq -2 \\ & \qquad\qquad -0.25x+y \leq 2 \\ & \qquad\qquad x+0.5y \leq 8 \\ & \qquad\qquad x-2y \leq 2 \\ & \qquad\qquad -y \leq 0. \end{array}$$

Considering the KKT optimality conditions of the inner problem and introducing the binary variables a_i, the bilevel problem becomes:

$$\min_{x,y,u} \quad F(x,y) = x + y$$

s.t.

$$\left.\begin{array}{l} -x \leq 0 \\ -x - 0.5y \leq -2 \\ -0.25x + y \leq 2 \\ x + 0.5y \leq 8 \\ x - 2y \leq 2 \\ -y \leq 0 \end{array}\right\} \text{Feasibility Constraints}$$

$$\left.\begin{array}{l} a_1(-x - 0.5y + 2) = 0 \\ a_2(-0.25x + y - 2) = 0 \\ a_3(x + 0.5y - 8) = 0 \\ a_4(x - 2y - 2) = 0 \\ a_5(-y) = 0 \end{array}\right\} \text{Complementarity Constraints}$$

$$-0.5u_1 + u_2 + 0.5u_3 - 2u_4 - u_5 = 1 \ \} \ \text{Stationarity Constraint}$$

$$\left.\begin{array}{l} u_i \leq Ma_i, \ i = 1,..,5 \\ a_i \leq Mu_i, \ i = 1,..,5 \end{array}\right\} \text{Logical Constraints}$$

$$a_i = \{0 - 1\}, \ u_i \geq 0.$$

Before starting the iterations, problem (PB^i) is formulated and solved for each constraint $i = 1,..,5$ which yields the following lower bounds for the constraints:

$$LB_1 = -6, \ LB_2 = -3, \ LB_3 = -6, \ LB_4 = -8, \ LB_5 = -4.$$

Iteration 1: Consider a starting point of $a_i = 1, i = 1,..,5$ which corresponds to all inner constraints being active. The primal problem can then be written as:

$$\min_{x,y} \quad F(x,y) = x + y$$

s.t.

$$\begin{array}{l} -x \leq 0 \\ -x - 0.5y = -2 \\ -0.25x + y = 2 \\ x + 0.5y = 8 \\ x - 2y = 2 \\ -y = 0. \end{array}$$

This problem is infeasible and hence, the following relaxed primal problem is formulated and solved:

$$\min_{x,y,s_i^+,s_i^-} \quad \sum_{i=1}^{5}(s_i^+ + s_i^-)$$

s.t.

$$\begin{aligned}
&-x \leq 0\\
&-x - 0.5y + s_1^+ - s_1^- = -2\\
&-0.25x + y + s_2^+ - s_2^- = 2\\
&x + 0.5y + s_3^+ - s_3^- = 8\\
&x - 2y + s_4^+ - s_4^- = 2\\
&-y + s_5^+ - s_5^- = 0
\end{aligned}$$

The Lagrange cut formulated for this problem has the following form:

$$6a_1 + 3a_2 + 6a_3 + 2a_4 + 2a_5 - 10 \leq 0.$$

The relaxed dual subproblem is :

$$\min_{\boldsymbol{a},\boldsymbol{u}} \quad \mu_B$$

s.t.

$$\begin{aligned}
&6a_1 + 3a_2 + 6a_3 + 2a_4 + 2a_5 - 10 \leq 0\\
&-0.5u_1 + u_2 + 0.5u_3 - 2u_4 - u_5 = 1\\
&u_i \leq Ma_i,\ i = 1,..,5\\
&a_i \leq Mu_i,\ i = 1,..,5\\
&a_2 + a_3 \geq 1\\
&\sum_{i=1}^{5} a_i \leq 1\\
&\mu_B \leq U
\end{aligned}$$

where U is a very large positive number. The solution of this problem is $\mu_B = U$ and $a_3 = 1$ that makes the third constraint active.

Iteration 2: For $a_3 = 1$, the primal problem is:

$$\min_{x,y} \quad F(x,y) = x + y$$

s.t.

$$\begin{aligned}
&-x \leq 0\\
&-x - 0.5y \leq -2\\
&-0.25x + y \leq 2\\
&x + 0.5y = 8\\
&x - 2y \leq 2\\
&-y \leq 0.
\end{aligned}$$

The primal is feasible and its solution yields:

$$x = 7.2, \quad y = 1.6, \quad Z^2 = 8.8.$$

The Lagrange function formulated from the second primal problem is:

$$L(\boldsymbol{a}) = 1.6 + 7.2a_3$$

and the relaxed dual subproblem:

$$\begin{aligned}
&\min_{\boldsymbol{a},\boldsymbol{u}} \quad \mu_B \\
\text{s.t.} \quad & \\
&\mu_B \geq 1.6 + 7.2a_3 \\
&6a_1 + 3a_2 + 6a_3 + 2a_4 + 2a_5 - 10 \leq 0 \\
&-0.5u_1 + u_2 + 0.5u_3 - 2u_4 - u_5 = 1 \\
&u_i \leq Ma_i, \; i = 1, ..., 5 \\
&a_i \leq Mu_i, \; i = 1, ..., 5 \\
&a_2 + a_3 \geq 1 \\
&\sum_{i=1}^{5} a_i \leq 1 \\
&\mu_B \leq 8.8 \\
&\mu_B \geq L.
\end{aligned}$$

Its solution is:

$$\mu_B = 1.6, \quad a_2 = 1.$$

Thus, after the second iteration, $P^{UBD} = 8.8, R^{LBD} = 1.6$ and $a_2 = 1$.

Iteration 3: The primal problem is solved for $a_2 = 1$. The solution of this problem gives:

$$x = 0.889, \quad y = 2.222, \quad Z^3 = 3.111$$

The Lagrange function from the third primal problem is:

$$L(\boldsymbol{a}) = 1.777 + 1.333a_2$$

and the relaxed dual subproblem:

$$\min_{\boldsymbol{a},\boldsymbol{u}} \quad \mu_B$$

s.t.

$$\mu_B \geq 1.777 + 1.333a_2$$
$$\mu_B \geq 1.6 + 7.2a_3$$
$$6a_1 + 3a_2 + 6a_3 + 2a_4 + 2a_5 - 10 \leq 0$$

$$-0.5u_1 + u_2 + 0.5u_3 - 2u_4 - u_5 = 1$$

$$u_i \leq Ma_i, \; i = 1, ..., 5$$
$$a_i \leq Mu_i, \; i = 1, ..., 5$$

$$a_2 + a_3 \geq 1$$
$$\sum_{i=1}^{5} a_i \leq 1$$

$$\mu_B \leq 3.111$$
$$\mu_B \geq 1.6.$$

Its solution is:

$$\mu_B = 3.111, \quad a_2 = 1$$

and after the third iteration, $P^{UBD} = 1.6, R^{LBD} = 1.6$, $a_2 = 1$ and $(x, y) =$ (0.889,2.222) which corresponds to the global optimum.

5.3.2 *Computational Experience*

The modified GOP algorithm has been coded in C and tested for a series of small literature example problems. The results are summarized in Table 5.1.

A number of randomly generated problems with the same characteristics as in Bard and Moore (1990), and Hansen et al. (1992) with 40% and 33% density, have also been considered. The results are summarized in Table 5.2. All computations are performed using CPLEX for the solution of linear problems using a HP-730. Performance measures include CPU time and the number of iterations required to obtain the global solution. The main parameters considered are the number of the follower's constraints and the numbers n_x, n_y of the leader's and the follower's variables. Problems involving 12-17 constraints and 40-50 variables were solved.

As expected the CPU time and the number of iterations increases with the size of the problem. Notice however, that the number of required iterations

EXAMPLES	n_x	n_y	Outer Constraints	Inner Constraints	Iterations	CPU time (s)
EX1	2	3	2	6	5	0.59
EX2	1	1	1	7	2	0.11
EX3	1	1	1	6	3	0.29
EX4	6	3	6	10	3	0.75
EX5	1	1	1	5	3	0.29
EX6	1	2	1	4	2	0.16
EX7	1	1	1	4	3	0.23
EX8	1	1	1	4	3	0.22
EX9	1	1	1	5	3	0.29
EX10	1	2	2	4	2	0.16
EX11	2	3	3	6	5	0.82

Table 5.1. Results for small linear examples

remains relatively low. Also, as it is shown in Table 5.2, large differences in computational effort are observed between problems of the same size (e.g., for the case of 26 outer and 14 inner variables, computation times of 867 and 81 sec. have been reported for two different examples).

5.4 BILEVEL LINEAR-QUADRATIC AND BILEVEL QUADRATIC-QUADRATIC PROBLEMS

In this section the approach is extended to the bilevel linear-quadratic and the bilevel quadratic-quadratic programming problems of the following general form:

$$\begin{aligned} &\min_{\boldsymbol{x}} \quad F(\boldsymbol{x},\boldsymbol{y}) \\ &s.t. \\ &\boldsymbol{y} \in \left\{ \begin{array}{ccc} & \min\limits_{\boldsymbol{y}} \ f(\boldsymbol{x},\boldsymbol{y}) & \\ s.t. & \boldsymbol{A}\boldsymbol{x}+\boldsymbol{B}\boldsymbol{y} & \leq \ \boldsymbol{b} \\ & \boldsymbol{x} \geq \boldsymbol{0} & \end{array} \right\} \end{aligned} \tag{5.15}$$

where $F(\boldsymbol{x},\boldsymbol{y})$ is a convex function of $\boldsymbol{x}$ and $\boldsymbol{y}$, and $f(\boldsymbol{x},\boldsymbol{y}) = \boldsymbol{d}_2^T\boldsymbol{y} + \boldsymbol{x}^T\boldsymbol{Q}_1^2\boldsymbol{y} + \boldsymbol{y}^T\boldsymbol{Q}_2^2\boldsymbol{y}$. For the sake of simplicity, it is assumed that $F(\boldsymbol{x},\boldsymbol{y}) = \boldsymbol{c}_1^T\boldsymbol{x} + \boldsymbol{d}_1^T\boldsymbol{y}$. It can easily be shown, however, that the following analysis is valid for any convex form of $F(\boldsymbol{x},\boldsymbol{y})$. It is also assumed that $f(\boldsymbol{x},\boldsymbol{y})$ is a convex quadratic function. Then, the KKT conditions for the inner problem are both necessary

DENSITY	n_x	n_y	Inner Constraints	Iterations	CPU time (s)
40%	28	12	12	7	8.4
40%	28	12	12	57	121.0
40%	28	12	12	9	8.0
40%	28	12	12	14	6.8
40%	26	14	14	26	81.1
40%	26	14	14	88	867.6
40%	25	15	15	12	42.8
40%	25	15	15	20	52.1
40%	24	17	17	5	34.0
40%	30	15	15	36	129.8
40%	35	15	15	26	282.3
40%	35	15	15	85	455.3
33%	27	13	13	54	249.6
33%	27	13	13	17	52.6
33%	27	13	13	21	98.8

Table 5.2. Computational Results for randomly generated linear problems

and sufficient for inner optimality which preserves the equivalence shown below:

$$\begin{aligned}
\min_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{u}} \quad & F(\boldsymbol{x},\boldsymbol{y}) = \boldsymbol{c}_1^T\boldsymbol{x} + \boldsymbol{d}_1^T\boldsymbol{y} \\
\text{s.t.} \quad & \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{y} \leq \boldsymbol{b} \\
& 2\boldsymbol{y}^T\boldsymbol{Q}_2^2 + \boldsymbol{x}^T\boldsymbol{Q}_1^2 + \boldsymbol{u}^T\boldsymbol{B}_2 + \boldsymbol{d}_2^T = \boldsymbol{0} \\
& u_i(\boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{y} - \boldsymbol{b})_i = 0, \quad i = 1,..,m \\
& \boldsymbol{x} \in X, \ \boldsymbol{y} \in Y, \ \boldsymbol{u} \geq \boldsymbol{0}.
\end{aligned}$$

Introducing the set of 0-1 variables, a_i, results in following equivalent formulation:

$$\begin{aligned}
\min_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{a},\boldsymbol{u}} \quad & F(\boldsymbol{x},\boldsymbol{y}) = \boldsymbol{c}_1^T\boldsymbol{x} + \boldsymbol{d}_1^T\boldsymbol{y} \\
\text{s.t.} \quad & \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{y} \leq \boldsymbol{b} \\
& 2\boldsymbol{y}^T\boldsymbol{Q}_2^2 + \boldsymbol{x}^T\boldsymbol{Q}_1^2 + \boldsymbol{u}^T\boldsymbol{B}_2 + \boldsymbol{d}_2^T = \boldsymbol{0} \\
& a_i(\boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{y} - \boldsymbol{b})_i = 0, \quad i = 1,..,m \\
& u_i \leq Ma_i, \quad i = 1,..,m \\
& a_i \leq Mu_i, \quad i = 1,..,m \\
& \boldsymbol{x} \in X, \ \boldsymbol{y} \in Y, \ a_i = \{0-1\}, \ u_i \geq 0, \ i = 1,..,m.
\end{aligned}$$

As in the linear case, the variables can be partitioned into $Y = (\boldsymbol{a}, \boldsymbol{u})$, and $X = (\boldsymbol{x}, \boldsymbol{y})$. Then, for fixed $Y = Y^k$ the primal problem becomes:

$$\begin{aligned} \min_{\boldsymbol{x},\boldsymbol{y}} \quad & F(\boldsymbol{x},\boldsymbol{y}) = \boldsymbol{c}_1^T\boldsymbol{x} + \boldsymbol{d}_1^T\boldsymbol{y} \\ \text{s.t.} \quad & \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{y} \leq \boldsymbol{b} \\ & 2y^T\boldsymbol{Q}_2^2 + \boldsymbol{x}^\top\boldsymbol{Q}_1^2 + \boldsymbol{u}^{kT}\boldsymbol{B}_2 + \boldsymbol{d}_2^T = \boldsymbol{0} \\ & a_i^k(\boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{y} - \boldsymbol{b})_i = 0, \quad i = 1,..,m \\ & \boldsymbol{x} \in X, \ \boldsymbol{y} \in Y. \end{aligned}$$

The Lagrange function of this problem becomes:

$$\begin{aligned} L(\boldsymbol{x},\boldsymbol{y},\boldsymbol{a},\boldsymbol{u},\mu^k,\lambda^k) \quad = \quad & \boldsymbol{c}_1^T\boldsymbol{x} + \boldsymbol{d}_1^T\boldsymbol{y} + \sum_{i=1}^{m}[(\mu_i^k + \lambda_i^k a_i)\cdot(\boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{y} - \boldsymbol{b})_i] \\ & + \sum_{i=1}^{n_y}\nu_i^k[2\boldsymbol{y}^T\boldsymbol{Q}_2^2 + \boldsymbol{x}^T\boldsymbol{Q}_1^2 + \boldsymbol{u}^T\boldsymbol{B}_2 + \boldsymbol{d}_2^T]_i. \end{aligned}$$

Separating the terms in $\boldsymbol{x}$ and $\boldsymbol{y}$, this can be rewritten as:

$$\begin{aligned} L(\boldsymbol{x},\boldsymbol{y},\boldsymbol{a},\boldsymbol{u},\mu^k,\lambda^k) \quad = \quad & [\boldsymbol{c}_1 + \sum_{i=1}^{m}(\mu_i^k + \lambda_i^k u_i)\boldsymbol{A}_i + \sum_{i=1}^{n_y}\nu_i^k\boldsymbol{Q}_{1_i}^2]^T\boldsymbol{x} \\ & + [\boldsymbol{d}_1 + \sum_{i=1}^{m}(\mu_i^k + \lambda_i^k u_i)\boldsymbol{B}_i + \sum_{i=1}^{n_y}2\nu_i^k\boldsymbol{Q}_{2_i}^2]^T\boldsymbol{y} \\ & - \sum_{i=1}^{m}(\mu_i^k + \lambda_i^k u_i)b_i + \sum_{i=1}^{n_y}\nu_i^k d_i^2. \end{aligned}$$

Using the KKT gradient conditions, the Lagrange function can be reduced to

$$\begin{aligned} L(\boldsymbol{x},\boldsymbol{y},\boldsymbol{a},\boldsymbol{u},\mu^k,\lambda^k,\nu^k) \quad = \quad & \sum_{i=1}^{m}\lambda_i^k(a_i - a_i^k)(\boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{y} - \boldsymbol{b})_i \\ & + \sum_{i=1}^{n_y}\nu_i^k(u_i - u_i^k)\boldsymbol{B}_i + (\boldsymbol{c}_1^T\boldsymbol{x} + \boldsymbol{d}_1^T\boldsymbol{y})^k. \end{aligned}$$

Then, it is obvious that Property 5.2.1 holds:

$$\begin{aligned} \min_{\boldsymbol{x},\boldsymbol{y}} \ L(\boldsymbol{x},\boldsymbol{y},\boldsymbol{a},\boldsymbol{u},\mu^k,\lambda^k,\nu^k) \quad \geq \quad & \sum_{i=1}^{m}\lambda_i^k(a_i - 1)(\boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{y} - \boldsymbol{b})_i^L \\ & + \sum_{i=1}^{n_y}\nu_i^k(u_i - u_i^k)\boldsymbol{B}_i + (\boldsymbol{c}_1^T\boldsymbol{x} + \boldsymbol{d}_1^T\boldsymbol{y})^k \end{aligned}$$

and consequently only one relaxed dual subproblem has to be solved per iteration.

Remark 5.4.1 Since the stationary conditions are functions of $\boldsymbol{x}$ and $\boldsymbol{y}$ variables, they appear to both primal and relaxed-dual subproblems. Moreover, for the case of quadratic outer objective $F(\boldsymbol{x}, \boldsymbol{y})$ the primal problem corresponds to a nonlinear programming problem. However, under the convexity assumptions it can be solved using a conventional local NLP solver.

5.4.1 Computational Studies

In this section, four examples are presented to highlight the performance of the proposed approach.

Example 5.4.1 This example is taken from Bard (1988).

$$\begin{aligned}
&\min_{x} \quad F(x,y) = (x-5)^2 + (2y+1)^2 \\
&\text{s.t.} \quad -x \leq 0 \\
&\qquad \min_{y} \quad f(x,y) = (y-1)^2 - 1.5xy \\
&\qquad \text{s.t.} \quad -3x + y \leq -3 \\
&\qquad\qquad x - 0.5y \leq 4 \\
&\qquad\qquad x + y \leq 7 \\
&\qquad\qquad -y \leq 0.
\end{aligned}$$

There exist two local minima at (1,0) and (5,2). The modified GOP global optimization approach requires 8 iterations to reach the global minimum (1,0); $F = 17, f = 1$, within 3.3 CPU sec. using GAMS/MINOS for the solution of nonlinear primal subproblem and GAMS/SCICONIC for mixed integer relaxed-dual subproblem.

Example 5.4.2 This example is taken from Shimizu and Aiyoshi (1981).

$$\begin{aligned}
&\min_{x} \quad F(x,y) = x^2 + (y-10)^2 \\
&\text{s.t.} \quad x \leq 15 \\
&\qquad -x + y \leq 0 \\
&\qquad -x \leq 0 \\
&\qquad \min_{y} \quad f(x,y) = (x + 2y - 30)^2 \\
&\qquad \text{s.t.} \quad x + y \leq 20 \\
&\qquad\qquad 0 \leq y \leq 20.
\end{aligned}$$

The modified GOP algorithm identified the global solution (x, y) =(10,10); $F = 1000, f = 0$, in only two iterations within 0.12 CPU sec. on a HP-730 using GAMS/MINOS for the solution of nonlinear primal and GAMS/CPLEX for the MILP relaxed-dual subproblems.

Example 5.4.3 This example is also from Shimizu and Aiyoshi (1981).

$$
\begin{aligned}
\min_{x} \quad & F(x,y) = (x_1 - 30)^2 + (x_2 - 20)^2 - 20y_1 + 20y_2 \\
\text{s.t.} \quad & x_1 + 2x_2 \geq 30 \\
& x_1 + x_2 \leq 25 \\
& x_2 \leq 15 \\
& \min_{y} \quad f(x,y) = (x_1 - y_1)^2 + (x_2 - y_2)^2 \\
& \text{s.t.} \quad 0 \leq y_1 \leq 10 \\
& \qquad 0 \leq y_2 \leq 10.
\end{aligned}
$$

For this example two iterations were required for the modified GOP to find the global solution (x_1, x_2, y_1, y_2)=(20,5,10,5); $F = 225, f = 100$, in 0.12 CPU sec.

Note that for examples 2 and 3, the reported penalty method of Shimizu and Aiyoshi (1981) converges to near optimal solution as the penalty coefficient (r) decreases. Also, ill-conditioning problems were observed for small values of r.

Example 5.4.4 This example is taken from Aiyoshi and Shimizu (1984).

$$
\begin{aligned}
\min_{x} \quad & F(x,y) = 2x_1 + 2x_2 - 3y_1 - 3y_2 - 60 \\
\text{s.t.} \quad & x_1 + x_2 + y_1 - 2y_2 - 40 \leq 0 \\
& 0 \leq x_1 \leq 50 \\
& 0 \leq x_2 \leq 50 \\
& \min_{y} \quad f(x,y) = (y_1 - x_1 + 20)^2 + (y_2 - x_2 + 20)^2 \\
& \text{s.t.} \quad -x_1 + 2y_1 \leq -10 \\
& \qquad -x_2 + 2y_2 \leq -10 \\
& \qquad -10 \leq y_1 \leq 20 \\
& \qquad -10 \leq y_2 \leq 20.
\end{aligned}
$$

The implementation of the modified GOP algorithm for this problem leads to the global minimum solution (x_1, x_2, y_1, y_2)=(0,0,-10,-10), $F = 0, f = 200$, after 8 iterations within 0.8 CPU s using GAMS/MINOS for the solution of LP primal and GAMS/CPLEX for the MILP relaxed-dual subproblems. Note that the penalty solution method of Aiyoshi and Shimizu fails to identify the global optimum, converging to the local minimum (x_1, x_2, y_1, y_2)=(25,30,5,10) F=5, f=100.

Summary

This chapter presented the global optimization approach GOP for bilevel linear and quadratic programming problems. The exploitation of the special structure of the problem results in simplified primal and relaxed dual subproblems. In particular, the primal problem is a single linear/convex optimization problem (corresponding to an active set of the follower's constraints) whereas the relaxed dual problem is solved through *one* single subproblem. The simplified primal and relaxed dual problems result in a modified GOP algorithm that is computationally efficient.

6 THE GOP APPROACH IN PHASE AND CHEMICAL EQUILIBRIUM PROBLEMS

This chapter introduces the reader to the global optimization of phase and chemical reaction equilibrium problems through the GOP algorithm studied by McDonald and Floudas (1994), (1995a), and (1995b). Two main classes of phase and chemical equilibrium problems are addressed : (a) the minimization of the Gibbs free energy and (b) the tangent plane stability criterion. Section 6.1 presents the formulation of minimization of the Gibbs free energy problem. Section 6.2 introduces the theoretical analysis for the application of the GOP approach using the NRTL activity coefficient model. Section 6.3 presents an illustrative example and a difficult phase equilibrium problem. Section 6.4 introduces the reader to the tangent plane stability problem and its mathematical formulation. Section 6.5 discusses the theoretical developments for applying the GOP to the tangent plane stability criterion, and finally section 6.6 presents an illustrative example.

6.1 INTRODUCTION AND FORMULATION OF MINIMIZING THE GIBBS FREE ENERGY

In this section, we present a general outline of the minimization of the Gibbs free energy problem. The focus is on systems that attain equilibrium states under conditions of constant temperature and pressure, where the global minimum value of the Gibbs free energy describes the true equilibrium state. The problem may be stated as follows:

C. A. Floudas, *Deterministic Global Optimization*
© Springer Science+Business Media Dordrecht 2000

Given C components participating in up to P potential phases under isothermal and isobaric conditions find the mole vector $\boldsymbol{n}$ that minimizes the value of the Gibbs free energy while also satisfying the appropriate material balance constraints.

The set of components is represented by the index set $C = \{i\}$ and the elements that constitute these components are given by $E = \{e\}$. The set of phases is denoted by $P = \{k\}$ where it is composed of vapor and liquid phases, labeled P_V and P_L respectively, so that $P \equiv P_V \cup P_L$. The liquid phase mole fractions are defined as $\boldsymbol{x} \equiv \{x_i\}$, and the vapor phase mol fractions are denoted $\boldsymbol{y} \equiv \{y_i\}$.

6.1.1 Molar Gibbs function calculated using fugacities:

The molar Gibbs free energy function is defined in terms of the chemical potentials of the individual components, which can be expressed in terms of the fugacities as follows:

$$\begin{aligned} g(\boldsymbol{x}) &= \sum_{i \in C} x_i \mu_i \\ &= \sum_{i \in C} x_i \left\{ \mu_i^0 + RT \frac{\hat{f}_i}{f_i^0} \right\} \end{aligned} \tag{6.1}$$

where μ_i and $\hat{f}_i$ are defined as the chemical potential and fugacity of component i in the mixture respectively, with μ_i^0 and f_i^0 equal to the corresponding values of pure component i at the standard state. The liquid phase can be modeled through the use of activity coefficients where the fugacity ratio is expressed as:

$$\frac{\hat{f}_i^L}{f_i^{L,0}} = \gamma_i^L x_i^L \tag{6.2}$$

where x_i^L denotes the mole fraction of species i in the liquid phase, and γ_i^L is the corresponding activity coefficient at the system temperature and pressure. If the convention that $\ln \gamma_i \to 0$ as $x_i \to 1$ for the activity coefficient is assumed, then the standard state is conveniently taken as the fugacity of the component in its pure state at the temperature and pressure of the system. For low pressure systems, the variance of this quantity with pressure is often not significant.

The fugacity of the vapor phase can be expressed as a function of fugacity coefficients as follows:

$$\frac{\hat{f}_i^V}{f_i^{V,0}} = \phi_i^V y_i^V P \tag{6.3}$$

where ϕ_i^V is the partial fugacity coefficient, y_i^V represents the vapor phase mol fraction at a total system pressure P. The standard state for the vapor phase is taken as an ideal gas at unit fugacity at the system temperature where this quantity is usually equal to 1 atm.

6.1.2 Molar Gibbs function calculated using excess Gibbs energy:

For liquid phases, nonideality is often expressed through the *excess* functions which attempt to correlate the deviation of the system from ideality. There are several equations available for the description of nonideality. The molar Gibbs free energy is given by the sum of the ideal and excess Gibbs functions as follows:

$$\frac{g(\boldsymbol{x})}{RT} = \frac{g^I(\boldsymbol{x})}{RT} + \frac{g^E(\boldsymbol{x})}{RT}. \tag{6.4}$$

The molar Gibbs function for an ideal system is defined as follows:

$$\frac{g^I(\boldsymbol{x})}{RT} = \sum_{i \in C} x_i \frac{\bar{g}_i}{RT} + \sum_{i \in C} x_i \ln x_i \tag{6.5}$$

where $\bar{g}_i$ is the molar Gibbs free energy of pure component i referred to the standard state, R is the gas constant, and T is the temperature of the system. The convexity of the ideal molar Gibbs free energy is well known (e.g.,White *et al.*, 1958).

The excess Gibbs energy is described through the use of activity coefficients for liquid phases, and the two quantities are interrelated as follows :

$$\frac{g^E(\boldsymbol{x})}{RT} = \sum_{i \in C} x_i \ln \gamma_i \tag{6.6}$$

$$\text{and} \quad \ln \gamma_i = \left(\frac{\partial\, n^T g^E/RT}{\partial n_j} \right)_{T,P,n_{l \neq i,j}} \tag{6.7}$$

$$= \frac{g^E}{RT} - \sum_{j \neq i} x_j \left(\frac{\partial\, g^E/RT}{\partial x_j} \right)_{T,P,x_{l \neq i,j}} \tag{6.8}$$

where n^T is the total number of moles and n_i is the number of mols of component i present. These quantities are related to the mol fractions as $n_i = n^T x_i$. The index sets j and l also run over the set of components.

There are two possible ways of calculating the molar Gibbs free energy function and they are closely related. The first is to substitute Eqn. (6.2) or Eqn. (6.3) into Eqn. (6.1), and the second involves direct calculation of the Gibbs function from the excess Gibbs free energy using Eqn. (6.4). It should be noted that if the Gibbs free energy content of the elemental species is zero at the system temperature, then μ_i^0 or $\bar{g}_i$ is taken to equal the Gibbs free energy of formation of component i at the system temperature, denoted ΔG_i^f.

Thus, for a multicomponent, multiphase system, the criterion of equilibrium dictates that the total Gibbs free energy function attain its minimum, and this function is given by the summation of the molar Gibbs free energies for each

phase:

$$G(\boldsymbol{n}) = \sum_{k \in P} n^k g(\boldsymbol{x}^k) \tag{6.9}$$

where n^k is the total number of moles present in phase k and $g(\boldsymbol{x}^k)$ is the molar Gibbs free energy of phase k. The total number of mols in the system, denoted n^T is then calculated as $\sum_k n^k$. The global minimum of the total Gibbs free energy function in Eqn. (6.9) supplies the true equilibrium solution. This solution must satisfy some additional material constraints which are now described.

Material Balances

The objective function as described by Eqn. (6.9) must yield a solution that will satisfy the conservation of mass requirements. These can take either of two forms depending on whether reaction occurs in the system. They introduce a set of linear equality constraints into the formulation.

(a) *Elemental Constraints*

For simultaneous phase and chemical equilibrium where reaction does occur, conservation of the constituent atoms must be satisfied:

$$\sum_{i \in C} \sum_{k \in P} a_{ei} n_i^k = b_e \quad \forall \quad e \in E \tag{6.10}$$

where a_{ei} represents the number of gram-atoms of element e in component i, and b_e the total number of gram-atoms of element e in the system.

(b) *Mass Balance Constraints*

These constraints are required for those systems where no chemical reaction takes place, and thus conservation over the components need hold:

$$\sum_{k \in P} n_i^k = n_i^T \quad \forall \quad i \in C \tag{6.11}$$

where n_i^T is the total number of moles of component i in the beginning mixture.

For notational clarity, the material balance constraints for any system, reacting or non-reacting, will be written in the following general form:

$$\mathbf{A} \cdot \boldsymbol{n} - \mathbf{b} = 0 \tag{6.12}$$

where $\boldsymbol{n}$ represents the column vector of the component mole numbers, $\mathbf{A}$ is the appropriate elemental or compound abundance matrix, and $\mathbf{b}$ is the column vector of the total amounts of elements or compounds in the system.

Feasibility Constraints

Obviously a physically realizable solution requires that

$$0 \leq n_i^k \leq n^T \quad \forall \; i \in C \, , \; k \in P \tag{6.13}$$

where n^T is the total number of moles in the system.

Standard State Data:

It is appropriate to discuss assumptions that can be made in regard to the standard state data. If chemical reaction occurs in the system under investigation, then the Gibbs standard free energies of formation are kept as they are in Eqn. (6.9). For nonreacting systems, it is possible to remove certain constant terms from the objective function under two different scenarios:

Case I: If the system under consideration has no vapor phase postulated, and all the liquid phases share the same standard state, then it is possible to eliminate the Gibbs free energy of formation term due to the following observation:

$$\sum_{i\in C}\sum_{k\in P} n_i^k \Delta G_i^{k,f} = \sum_{i\in C} \Delta G_i^{L,f} \sum_{k\in P} n_i^k = \sum_{i\in C} \Delta G_i^{L,f} n_i^T = \kappa_I \tag{6.14}$$

where $\Delta G_i^{L,f}$ is the Gibbs formation energy for the liquid phase at the system temperature and pressure and κ_I is a constant. The modified objective function, labeled $\hat{G}_I(\boldsymbol{n})$, then becomes:

$$\min \quad \hat{G}_I(\boldsymbol{n}) = \sum_{i\in C}\sum_{k\in P} n_i^k \left\{ \frac{\hat{f}_i^k}{f_i^{k,0}} \right\} \equiv \frac{G(\boldsymbol{n}) - \kappa_I}{RT} \tag{6.15}$$

where $G(\boldsymbol{n})$ is defined by Eqn. (6.9).

Case II: If a vapor phase is present in the system, then the difference between the Gibbs formation energies of the gas and liquid states can usually be adequately approximated by a saturated pressure condensation term as follows:

$$\Delta G_i^{L,f} = \Delta G_i^{V,f} + RT \ln P_i^{SAT} \tag{6.16}$$

where P_i^{SAT} is the saturated vapor pressure for component i. Stull *et al.* (1969) note that Eqn. (6.16) is valid for vapor pressures under 2 or 3 atm, but may require the use of fugacity coefficients for systems that associate in the vapor phase. Defining $\kappa_{II} = \sum_i \Delta G_i^{V,f} n_i^T$, the modified objective function, labeled $\hat{G}_{II}(\boldsymbol{n})$, can now be written:

$$\begin{aligned} \min \quad \hat{G}_{II}(\boldsymbol{n}) &= \sum_{i\in C}\sum_{k\in P_L} n_i^k \ln P_i^{SAT} + \sum_{i\in C}\sum_{k\in P} n_i^k \left\{ \frac{\hat{f}_i^k}{f_i^{k,0}} \right\} \\ &\equiv \frac{G(\boldsymbol{n}) - \kappa_{II}}{RT} \end{aligned} \tag{6.17}$$

with $G(\boldsymbol{n})$ given by Eqn. (6.9). Thus, for phase equilibrium problems when only liquid phases are postulated, Eqns. (6.15) or (6.17) can be used to represent the Gibbs free energy function; if a vapor phase is present, then the function to be minimized is given by Eqn. (6.17). This minimizes the dependence of the reported solutions on the standard Gibbs free energy of formation data.

The complete formulation of the phase and chemical equilibrium for ideal vapor phases and liquid phases modelled via fugacity coefficients is given by

minimizing Eqn. (6.9) subject to the material balance constraints supplied by Eqn. (6.12), the feasibility constraints of Eqn. (6.13) and the definition of the fugacity coefficient model. The variables of the formulation are the mole numbers n_i^k. The mole fractions are eliminated by redefining them in terms of the mol numbers as $x_i = n_i / \sum_j n_j \; \forall \; i$ for all phases.

Remark 6.1.1 There are two important observations in regard to the optimization formulation:

- The constraint set is of small size and linear.
- The only nonlinearity appears in the objective function as $n_i \ln \hat{f}_i / f_i^0$.

If the system is ideal then any local solution will be the global one. However, the main difficulty is that due to the complex nature of the models used to predict fugacities, highly nonconvex functionalities result. This may lead to local or trivial solutions that are not true equilibrium solutions, and may lie far away from the correct optimal solution. The obtained solution will also be highly dependent on the chosen starting point.

6.2 APPLICATION OF THE GOP TO NRTL EQUATION

6.2.1 The NRTL Equation

In this section, the Gibbs free energy expression is analyzed for the case of an ideal vapor phase and liquid phases modeled using the NRTL activity coefficient expression. Renon and Prausnitz (1968) derived the following equation for the liquid-phase activity coefficient:

$$\ln \gamma_i = \frac{\sum\limits_{j \in C} \tau_{ji} \mathcal{G}_{ji} x_j}{\sum\limits_{j \in C} \mathcal{G}_{ji} x_j} + \sum_{j \in C} \frac{\mathcal{G}_{ij} x_j}{\sum\limits_{l \in C} \mathcal{G}_{lj} x_l} \left\{ \tau_{ij} - \frac{\sum\limits_{l \in C} \tau_{lj} \mathcal{G}_{lj} x_l}{\sum\limits_{l \in C} \mathcal{G}_{lj} x_l} \right\} \qquad \forall \; i \in C \quad (6.18)$$

where γ_i is the activity coefficient at mole fraction x_i, τ_{ij} and $\mathcal{G}_{ij}$ are non-symmetric binary interaction parameters. $\mathcal{G}_{ij}$ is based on another adjustable binary parameter α_{ij} (with $\alpha_{ii} = 0$) which does have the property of symmetry ($\alpha_{ij} = \alpha_{ji}$). Then $\mathcal{G}_{ij}$ is defined as:

$$\mathcal{G}_{ij} = \exp\left(-\alpha_{ij} \tau_{ij}\right)$$

τ_{ij} can be negative but $\mathcal{G}_{ij}$ is always positive. One important feature of the NRTL equation is its capability of representing liquid-liquid immiscibility for multi-component systems with only binary parameters. Eqn. (6.18) yields exactly the same expression for mole numbers as for mol fractions. Substitution of Eqn. (6.18) into Eqn. (6.2) yields the correct liquid phase fugacity term, after rewriting the mole fractions in terms of mol numbers. Eqn. (6.3) is assumed to define the vapor phase fugacity written in terms of the vapor mole

number variables. Substitution of the resultant vapor and liquid phase fugacity equations into the Gibbs free energy function leads to :

$$\min \hat{G}(\boldsymbol{n}) = \sum_{i\in C}\sum_{k\in P} n_i^k \left\{ \frac{\Delta G_i^{k,f}}{RT} + \ln \frac{n_i^k}{\sum_{j\in C} n_j^k} \right\}$$
$$+ \sum_{i\in C}\sum_{k\in P_L} n_i^k \left\{ \frac{\sum_{j\in C} \tau_{ji}\mathcal{G}_{ji} n_j^k}{\sum_{j\in C} \mathcal{G}_{ji} n_j^k} + \sum_{j\in C} \frac{\mathcal{G}_{ij} n_j^k}{\sum_{l\in C} \mathcal{G}_{lj} n_l^k} \left\{ \tau_{ij} - \frac{\sum_{l\in C} \tau_{lj}\mathcal{G}_{lj} n_l^k}{\sum_{l\in C} \mathcal{G}_{lj} n_l^k} \right\} \right\} \tag{6.19}$$

where $\hat{G}(\boldsymbol{n}) = G(\mathbf{n})/RT$ (i.e., dimensionless G). Note that the pressure term associated with the fugacity of the vapor phase has been incorporated into the Gibbs energy of formation term, that is, $\Delta G_i^{V,f} = \Delta G_i^{V,f} + RT \ln P$. This is done in order to collect the linear terms of the objective function.

6.2.2 *Analysis of the Gibbs free energy function:*

As given by Eqn. (6.18), $\hat{G}(\boldsymbol{n})$ is a complex and nonconvex expression. The following property leads to a major simplification :

Property 6.2.1 *For each phase $k \in P$, the following relation is true:*

$$\sum_{i\in C} n_i^k \left\{ \frac{\sum_{j\in C} \tau_{ji}\mathcal{G}_{ji} n_j^k}{\sum_{j\in C} \mathcal{G}_{ji} n_j^k} \right\} - \sum_{i\in C} n_i^k \left\{ \sum_{j\in C} \frac{\mathcal{G}_{ij} n_j^k}{\sum_{l\in C} \mathcal{G}_{lj} n_l^k} \frac{\sum_{l\in C} \tau_{lj}\mathcal{G}_{lj} n_l^k}{\sum_{l\in C} \mathcal{G}_{lj} n_l^k} \right\} = 0 \tag{6.20}$$

Proof: See McDonald and Floudas (1994).

This property reduces the complexity of Eqn. (6.18) greatly, and it brings the advantage of having bilinear, rather than trilinear, fractional functions in the expression for the objective function.

Property 6.2.2 *Let ρ_i be positive parameters defined $\forall\ i$. Define the real-valued function $f_i(\mathbf{n})$ with $\mathbf{n} > 0$ as follows:*

$$f_i(\mathbf{n}) = n_i \ln \frac{n_i}{\sum_j \rho_j n_j}$$

then $f_i(\mathbf{n})$ is convex.

Proof: See McDonald and Floudas (1994).

Remark 6.2.1 Based on Property 6.2.2, if C_N^k is defined as follows:

$$C_N^k = \sum_{i\in C} n_i^k \left\{ \frac{\Delta G_i^{k,f}}{RT} + \ln \frac{n_i^k}{\sum_{j\in C} n_j^k} \right\} \qquad \forall\, k \in P$$

then the quantity $\sum_k \mathcal{C}_N^k$ is convex, since it is a summation of individually linear and convex terms.

This means that the objective function can now be written as a combination of a convex portion, and a nonconvex portion:

$$\min \hat{G}(\boldsymbol{n}) \;=\; \sum_{k\in P} \mathcal{C}_N^k + \sum_{i\in C}\sum_{k\in P_L} n_i^k \left\{ \sum_{j\in C} \frac{\mathcal{G}_{ij}\tau_{ij}n_j^k}{\sum_{l\in C}\mathcal{G}_{lj}n_l^k} \right\}. \tag{6.21}$$

The nonconvexities of Eqn. (6.21) now exist in the term to the right of the plus sign.

6.2.3 Transformations and Partitioning

It is now convenient to introduce new variables in order to change the nature of the nonconvexities in the objective function. Having augmented the variable set in this way, it is then partitioned into two variable subsets, so that *Conditions (A)* of the GOP algorithm are satisfied. If the following new variables are introduced:

$$\Psi_i^k \;=\; \frac{n_i^k}{\sum_{j\in C}\mathcal{G}_{ji}n_j^k} \qquad \forall\; i\in C\;,\; k\in P_L \tag{6.22}$$

then the transformed objective function becomes:

$$\min \hat{G}(\boldsymbol{\Psi},\boldsymbol{n}) \;=\; \sum_{k\in P} \mathcal{C}^k + \sum_{i\in C}\sum_{k\in P_L} n_i^k \left\{ \sum_{j\in C} \mathcal{G}_{ij}\tau_{ij}\Psi_j^k \right\}. \tag{6.23}$$

This is now subject to the transformation constraint Eqn. (6.22) in addition to the material balance constraints as defined by Eqn. (6.12) so that the nonconvex formulation for the minimization of the Gibbs free energy using the NRTL equation, denoted (G-N), is given as:

$$\left.\begin{array}{lll}
\min & \hat{G}(\boldsymbol{\Psi},\boldsymbol{n}) & \\
s.t. & \Psi_i^k \cdot \left\{ \sum_{j\in C}\mathcal{G}_{ji}n_j^k \right\} = n_i^k & \forall\; i\in C\;,\; k\in P_L \\
 & \mathbf{A}\,\boldsymbol{n} - \mathbf{b} \;=\; \mathbf{0} & \\
\text{with} & Y \equiv \{n_i^k : 0 \le n_i^k \le n^T \quad \forall\, i\in C\;,\; \forall\, k\in P\} &
\end{array}\right\} \text{(G-N)}$$

where $\hat{G}(\boldsymbol{\Psi},\boldsymbol{n})$ is given by Eqn. (6.23). Note that Eqn. (6.22) has been reformulated so as to be of bilinear form. The objective is to partition the variable set

into two subsets so that if either of these subsets is held constant, an optimization problem with simpler structure remains. An examination of Eqn. (6.23) leads to the conclusion that the obvious partition of variables is that in which the $\boldsymbol{y}$ variable set contains the mole vector, with the $\boldsymbol{x}$ variable set containing the new transformed variables:

$$\boldsymbol{y} \leftarrow \{n_i^k\} \equiv \boldsymbol{n} \qquad \boldsymbol{x} \leftarrow \{\Psi_i^k\} \equiv \boldsymbol{\Psi}. \tag{6.24}$$

Notice that if the mole number variable set is held constant, a linear objective function results. On the other hand, if the transformed variable set is held constant, a convex objective function is obtained. The equality constraints are of bilinear form and so will yield linear terms if either of the subsets is held constant. It should be noted that the material balance equations depend on the mole numbers alone, and therefore have no interaction with the $\boldsymbol{x}$ variable set. Thus, *Conditions (A)* of the GOP algorithm are satisfied. The nonconvexities of the objective function have been changed, and additional *bilinear* constraints have been introduced.

6.2.4 The Primal Problem

The primal problem is defined as the subproblem that results when the $\boldsymbol{y}$ variable set is held fixed. In what follows, overbars on variables represent the values that are held fixed during solution of any given primal problem, which is defined as follows:

$$\left.\begin{aligned} \min \quad & \hat{G}(\boldsymbol{\Psi}, \bar{\boldsymbol{n}}) \equiv \sum_{k \in P} \bar{\mathcal{C}}^k + \sum_{i \in C} \sum_{k \in P_L} \bar{n}_i^k \left\{ \sum_{j \in C} \mathcal{G}_{ij} \tau_{ij} \Psi_j^k \right\} \\ s.t. \quad & \Psi_i^k \cdot \left\{ \sum_{j \in C} \mathcal{G}_{ji} \bar{n}_j^k \right\} = \bar{n}_i^k \qquad \forall\ i \in C\ ,\ k \in P_L \end{aligned}\right\} \tag{G-P}$$

$\bar{\boldsymbol{n}}$ represents the current value of the mole numbers (the $\boldsymbol{y}$ variable set).

Remark 6.2.2 The primal problem (G-P) is always feasible provided that the mole vector satisfies the material balance constraints which are functions of the $\boldsymbol{y}$ variables alone. Hence they can be carried directly to the relaxed dual subproblems. This is the reason the material balance constraints are not included in (G-P). Notice that (G-P) is merely a function evaluation as the $\boldsymbol{x}$ variable set is completely specified by $\bar{\boldsymbol{n}}$ through the set of equality constraints of (G-P).

It will be necessary to use the Karush-Kuhn-Tucker (KKT) conditions for the primal problem in proceeding sections. The Lagrangian as constructed from

the primal problem for fixed $\bar{\boldsymbol{n}}$ is given as:

$$\begin{aligned} L(\boldsymbol{\Psi}, \bar{\boldsymbol{n}}, \boldsymbol{\lambda}) &= \sum_{k \in P} \bar{\mathcal{C}}^k + \sum_{i \in C} \sum_{k \in P_L} \bar{n}_i^k \left\{ \sum_{j \in C} \mathcal{G}_{ij} \tau_{ij} \Psi_j^k \right\} \\ &+ \sum_{i \in C} \sum_{k \in P_L} \lambda_{\Psi_i^k} \left\{ \Psi_i^k \cdot \sum_{j \in C} \mathcal{G}_{ji} \bar{n}_j^k - \bar{n}_i^k \right\} \end{aligned} \tag{6.25}$$

where $\lambda_{\Psi_i^k}$ is the multiplier associated with the corresponding constraint that defines the x variable Ψ_i^k. The evaluation of the KKT conditions for the primal yields:

$$\nabla_{\Psi_i^k} L(\boldsymbol{\Psi}, \bar{\boldsymbol{n}}, \boldsymbol{\lambda}) = \sum_{j \in C} \mathcal{G}_{ji} \tau_{ji} \bar{n}_j^k + \lambda_{\Psi_i^k} \cdot \sum_{j \in C} \mathcal{G}_{ji} \bar{n}_j^k = 0 \quad \forall i \in C,\ k \in P_L \tag{6.26}$$

The Lagrange multipliers from the primal are then explicitly calculated as:

$$\lambda_{\Psi_i^k} = -\frac{\sum_{j \in C} \mathcal{G}_{ji} \tau_{ji} \bar{n}_j^k}{\sum_{j \in C} \mathcal{G}_{ji} \bar{n}_j^k} \qquad \forall\, i \in C\ ,\ k \in P_L. \tag{6.27}$$

Remark 6.2.3 If a given phase k disappears so that $n_i^k = 0\ \forall\ i$, then the denominator of Eqn. (6.27) approaches 0. In this case, $\lambda_{\Psi_i^k} = 0\ \forall\ i$ for the phase k will ensure that the KKT conditions of the primal problem are satisfied, as the corresponding constraints of the primal are $\Psi_i^k \cdot 0 = 0\ \forall\, i$. This eliminates the problem of obtaining unbounded Lagrange multipliers. Thus, the multipliers from any primal problem are nonempty and bounded for all $\boldsymbol{y} \in Y$, a required condition to guarantee ϵ-global convergence. Clearly, the evaluation of the primal problem and the corresponding multipliers amounts to simple function evaluations.

Illustration: There are two components and one phase, so that n_1 and n_2 represent the variables of the problem. If the linear terms are neglected, then the convex portion $\mathcal{C}$ of the Gibbs free energy is defined as:

$$\mathcal{C} = n_1 \ln \frac{n_1}{n_1 + n_2} + n_2 \ln \frac{n_2}{n_1 + n_2}.$$

The Lagrange function can then be written as:

$$\begin{aligned} L^1(\boldsymbol{\Psi}, \boldsymbol{n}, \boldsymbol{\lambda}) &= \mathcal{C} + n_1 \Psi_2 \cdot \mathcal{G}_{12} \tau_{12} + n_2 \Psi_1 \cdot \mathcal{G}_{21} \tau_{21} \\ &+ \lambda_1 \cdot \{\Psi_1 [\, n_1 + \mathcal{G}_{21} n_2 \,] - n_1\} + \lambda_2 \cdot \{\Psi_2 [\, \mathcal{G}_{12} n_1 + n_2 \,] - n_2\} \end{aligned}$$

and its derivatives with respect to the x variables for fixed $\boldsymbol{n} = \bar{\boldsymbol{n}}$ are given as:

$$\begin{aligned} \nabla_{\Psi_1} L(\boldsymbol{\Psi}, \bar{\boldsymbol{n}}, \boldsymbol{\lambda}) &= \bar{n}_2 \cdot \mathcal{G}_{12} \tau_{12} + \lambda_1 \cdot [\, \bar{n}_1 + \mathcal{G}_{21} \bar{n}_2 \,] \\ \nabla_{\Psi_2} L(\boldsymbol{\Psi}, \bar{\boldsymbol{n}}, \boldsymbol{\lambda}) &= \bar{n}_1 \cdot \mathcal{G}_{21} \tau_{21} + \bar{\lambda}_2 \cdot [\, \mathcal{G}_{12} \bar{n}_2 + \bar{n}_2 \,]. \end{aligned}$$

6.2.5 *The Relaxed Dual Problem*

The primal problem establishes upper bounds on the solution. The relaxed dual subproblems supply lower bounds on the global solution. The derivation of the Lagrangian, is discussed next.

6.2.5.1 Derivation of the Lagrangian. The first step in deriving the Lagrangian is to separate and collect all the x variable terms so their interaction with the y variable set can be examined. For any fixed $\boldsymbol{\lambda} = \bar{\boldsymbol{\lambda}}$:

$$\begin{aligned} L(\boldsymbol{\Psi}, \boldsymbol{n}, \bar{\boldsymbol{\lambda}}) &= \sum_{i \in C} \sum_{k \in P_L} \Psi_i^k \left\{ \sum_{j \in C} \mathcal{G}_{ji} \tau_{ji} n_j^k + \bar{\lambda}_{\Psi_i^k} \cdot \sum_{j \in C} \mathcal{G}_{ji} n_j^k \right\} \\ &+ \sum_{k \in P} \mathcal{C}^k - \sum_{i \in C} \sum_{k \in P_L} n_i^k \bar{\lambda}_{\Psi_i^k}. \end{aligned} \tag{6.28}$$

It differs from the one of Eqn. (6.24) in that it is written for any y. By subtracting Eqn. Set (6.25) written for $\boldsymbol{\lambda} = \bar{\boldsymbol{\lambda}}$ from the terms within the curly braces of Eqn. (6.28) and collecting the terms in n_i^k together, the following expression is obtained:

$$\begin{aligned} L(\boldsymbol{\Psi}, \boldsymbol{n}, \bar{\boldsymbol{\lambda}}) &= \sum_{i \in C} \sum_{k \in P_L} \Psi_i^k \left\{ \sum_{j \in C} \mathcal{G}_{ji} \left[\tau_{ji} + \bar{\lambda}_{\Psi_i^k} \right] \left[n_j^k - \bar{n}_j^k \right] \right\} \\ &+ \sum_{k \in P} \mathcal{C}^k - \sum_{i \in C} \sum_{k \in P_L} n_i^k \bar{\lambda}_{\Psi_i^k}. \end{aligned} \tag{6.29}$$

Evaluating the gradients of the Lagrangian given by Eqn. (6.29) with respect to the $\boldsymbol{x}$ variables results in:

$$g_i^k(\boldsymbol{y}) = \nabla_{\Psi_i^k} L(\boldsymbol{\Psi}, \boldsymbol{n}, \bar{\boldsymbol{\lambda}}) = \sum_{j \in C} \mathcal{G}_{ji} \left[\tau_{ji} + \bar{\lambda}_{\Psi_i^k} \right] \cdot \left[n_j^k - \bar{n}_j^k \right] \quad \forall i \in C, k \in P_L. \tag{6.30}$$

These are the *qualifying* constraints written in terms of the $\boldsymbol{y}$ variables and describe the fundamental nature of the interaction of the two variable subsets. Notice that each $\boldsymbol{x}$ variable multiplies a *summation* of $\boldsymbol{y}$ variables in Eqn. (6.30), so that these constraints form hyperplanes that partition the $\boldsymbol{y}$ variable space.

Illustration – *continued*: Collecting the $\boldsymbol{x}$ variables of $L^1(\boldsymbol{\Psi}, \boldsymbol{n}, \bar{\boldsymbol{\lambda}})$ yields:

$$\begin{aligned} L^2(\boldsymbol{\Psi}, \boldsymbol{n}, \bar{\boldsymbol{\lambda}}) &= C - n_1 \bar{\lambda}_1 - n_2 \bar{\lambda}_2 \\ &+ \Psi_1 \cdot \left\{ n_2 \mathcal{G}_{21} \tau_{21} + \bar{\lambda}_1 \left[n_1 + \mathcal{G}_{21} n_2 \right] \right\} + \Psi_2 \cdot \left\{ n_1 \mathcal{G}_{12} \tau_{12} + \bar{\lambda}_2 \left[\mathcal{G}_{21} n_1 + n_2 \right] \right\}. \end{aligned}$$

For any given primal problem, the KKT conditions yield $\nabla_{\Psi_i} L(\boldsymbol{\Psi}, \bar{\boldsymbol{n}}, \boldsymbol{\lambda}) = 0 \; \forall \; i$ and are clearly satisfied for $\boldsymbol{\lambda} = \bar{\boldsymbol{\lambda}}$. Subtracting these derivatives from the terms within the curly braces of $L^2(\boldsymbol{\Psi}, \boldsymbol{n}, \bar{\boldsymbol{\lambda}})$ allows the terms multiplying the

x variables to be written in the form $n_i^k - \bar{n}_i^k$. After collecting the terms in n_i together, the following result is obtained:

$$\begin{aligned} L^3(\boldsymbol{\Psi}, \boldsymbol{n}, \bar{\boldsymbol{\lambda}}) &= C - n_1\bar{\lambda}_1 - n_2\bar{\lambda}_2 \\ &+ \Psi_1 \cdot \{\bar{\lambda}_1 [\, n_1 - \bar{n}_1 \,] + \mathcal{G}_{21} [\, \tau_{21} + \bar{\lambda}_1 \,] [\, n_2 - \bar{n}_2 \,]\} \\ &+ \Psi_2 \cdot \{\mathcal{G}_{12} [\, \tau_{12} + \bar{\lambda}_2 \,] [\, n_1 - \bar{n}_1 \,] + \bar{\lambda}_2 [\, n_2 - \bar{n}_2 \,]\}. \end{aligned}$$

The qualifying constraints are then defined as:

$$\begin{aligned} \nabla_{\Psi_1} L(\boldsymbol{\Psi}, \boldsymbol{n}, \bar{\boldsymbol{\lambda}}) &= \bar{\lambda}_1 [\, n_1 - \bar{n}_1 \,] + \mathcal{G}_{21} [\, \tau_{21} + \bar{\lambda}_1 \,] [\, n_2 - \bar{n}_2 \,] \\ \nabla_{\Psi_2} L(\boldsymbol{\Psi}, \boldsymbol{n}, \bar{\boldsymbol{\lambda}}) &= \mathcal{G}_{12} [\, \tau_{12} + \bar{\lambda}_2 \,] [\, n_1 - \bar{n}_1 \,] + \bar{\lambda}_2 [\, n_2 - \bar{n}_2 \,]. \end{aligned}$$

The form of these constraints will be changed in the following.

6.2.5.2 Partitioning of the y variable space. The next important step is to obtain a much simpler set of partitioning hyperplanes. This is achieved by simply augmenting the set of $\boldsymbol{x}$ variables, so that each one of these new $\boldsymbol{x}$ variables will interact with a *single* $\boldsymbol{y}$ variable, rather than a summation of them. This augmented set of variables, denoted $\{\hat{\Psi}_{ij}^k\}$, is defined for each $\{i, k\} \in C \times P_L$ as follows:

$$\hat{\Psi}_{ij}^k = \Psi_i^k \qquad \forall\, j \in C. \tag{6.31}$$

The $\boldsymbol{x}$ variables are now allowed to appear within the innermost summation of Eqn. (6.29) to yield an equivalent Lagrangian defined as follows:

$$\begin{aligned} L(\hat{\boldsymbol{\Psi}}, \boldsymbol{n}, \bar{\boldsymbol{\lambda}}) &= \sum_{i \in C} \sum_{k \in P_L} \left\{ \sum_{j \in C} \hat{\Psi}_{ij}^k \left[\mathcal{G}_{ji} \left\{ \tau_{ji} + \bar{\lambda}_{\Psi_i^k} \right\} \right] \cdot \left[n_j^k - \bar{n}_j^k \right] \right\} \\ &+ \sum_{k \in P} \mathcal{C}^k + \sum_{i \in C} \sum_{k \in P_L} n_i^k \left\{ \sum_{j \in C} \mathcal{G}_{ij} \tau_{ij} \bar{\Psi}_j^k \right\}. \end{aligned} \tag{6.32}$$

Eqn. Set (6.27) has been used to modify the terms involving $\bar{\lambda}_{\Psi_i^k}$ to the right of the minus sign of Eqn. (6.29). This has been done solely to demonstrate that if $\boldsymbol{n} = \bar{\boldsymbol{n}}$, then the Lagrangian equals the objective function value supplied by the primal at $\bar{\boldsymbol{n}}$. This is a statement of strong duality theory. Eqn. (6.32) now supplies the new form of the qualifying constraints, labeled $\hat{g}_{ij}^k(\boldsymbol{y})$, obtained from the modified Lagrangian of Eqn. (6.32) as:

$$\hat{g}_{ij}^k(\boldsymbol{y}) = \nabla_{\hat{\Psi}_{ij}^k} L(\boldsymbol{x}, \boldsymbol{y}, \bar{\lambda}) = \mathcal{G}_{ji} \left[\tau_{ji} + \bar{\lambda}_{\Psi_i^k} \right] \cdot \left[n_j^k - \bar{n}_j^k \right] \forall i \in C, j \in C, k \in P_L. \tag{6.33}$$

Remark 6.2.4 Note that each qualifying constraint is now a function of a single $\boldsymbol{y}$ variable, with the important result that the partitioning hyperplanes

are now orthogonal to each other, and partition the $\boldsymbol{y}$ variable space into n-rectangles. The number of connected variables is given as:

$$\begin{aligned} N_{CV} &= \min\{\, N_{CV}^{X} \,,\, N_{CV}^{Y} \,\} \\ &= \min\left\{\, |C|^2 \cdot |P_L| \,,\, |C| \cdot |P_L| \,\right\} \\ &= |C| \cdot |P_L| \end{aligned}$$

where the braces signify the cardinalities of the appropriate sets. Therefore, it is evident that augmenting the set of x variables in such a way does *not* increase the number of connected variables. Also note about Eqn. (6.33) is that each qualifying constraint shares the same basic form, defined as $(n_j^k - \bar{n}_j^k)$. The only difference is the expression that premultiplies this term. These are constants that depend on the parameters of the NRTL model and information from the primal in the form of the Lagrange multipliers.

Eqn. (6.13) delineates the feasible region as an n-rectangle. The initial *parent* region is described by this n-rectangle, and its bounds are represented by $R\{\mathcal{L}^R, \mathcal{U}^R\}$, where $\mathcal{L}^R = \{\mathcal{L}_{n_i^k}^R\}$ and $\mathcal{U}^R = \{\mathcal{U}_{n_i^k}^R\}$ comprise the regional bounds for the variables $\{n_i^k\}$. Upon choosing an initial point $\{\bar{n}_i^k\}$, this parent n-rectangle is partitioned by N_{CV} orthogonal hyperplanes passing through $\{\bar{n}_i^k\}$, so that $2^{N_{CV}}$ new n-rectangles are created. Within each of these new n-rectangles, the sign of $(n_i^k - \bar{n}_i^k)$ will be constant $\forall \;\; i \in C \;,\; k \in P_L$. The bounds for each of these n-rectangles are described as the *box* bounds, denoted $B\{\mathcal{L}^B, \mathcal{U}^B\}$, with $\mathcal{L}^B = \{\mathcal{L}_{n_i^k}^B\}$ and $\mathcal{L}^B = \{\mathcal{U}_{n_i^k}^B\}$ representing the individual box bounds for the variables $\{n_i^k\}$. The set of all possible combinations of box bounds is denoted by CB, with its $2^{N_{CV}}$ members individually referred to as B_l. The parameter $s_{ik}^{B_l}$ is used to delineate each of these box regions and is defined over $C \times P_L \times CB$. It determines the partition of the $\boldsymbol{y}$ variable space for any given B_l as follows:

$$\left.\begin{array}{lll} \text{If} \;\; s_{ik}^{B_l} = +1 & \text{then} & n_i^k - \bar{n}_i^k \geq 0 \\ \text{If} \;\; s_{ik}^{B_l} = -1 & \text{then} & n_i^k - \bar{n}_i^k < 0 \end{array}\right\} \;\; \forall\, i \in C \;,\; k \in P_L .$$

Figure 6.1 shows how $s_{ik}^{B_l}$ is used to create these regional and box bounds at the first iteration for the case of 2 connected variables, with $C \equiv \{i_1, i_2\}$ and $P_L \equiv \{k_1\}$. The initial point generates 4 subdomains denoted B_1 through B_4.

This implies that it is possible to construct Lagrangians that validly underestimate the global solution in each of these n-rectangles, within which an individual relaxed dual subproblem is solved. If the solution is greater than the current best upper bound obtained from the primal problem, it may be fathomed (i.e. discarded); otherwise, it is added to the set of candidate lower bounds. The infimum of all such solutions supplies the point for the next iteration, where this single n-rectangle will again be partitioned into $2^{N_{CV}}$ n-rectangles to supply additional lower bounds on the final solution. In the context of Figure 6.1, suppose the infimum of the 4 lower bounds lies in B_2. At the next iteration, B_2 is divided into 4 regions, and so on.

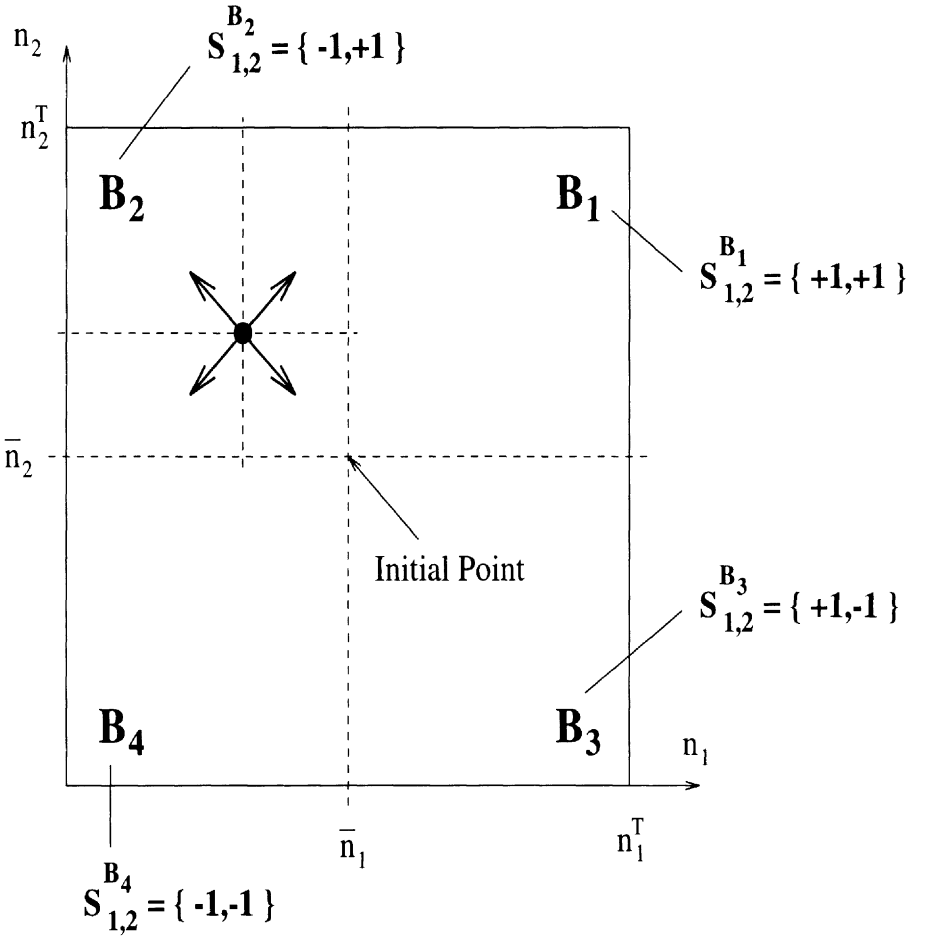


Figure 6.1. Example for two connected variables

A convenient way of describing this partitioning of the y variable space in the branch and bound approach is through the use of a tree structure. The starting point is represented by the *root* node, labeled R, and it generates $2^{N_{CV}}$ nodes at the first level of the tree. One of these leaf nodes becomes the next iteration node, in turn generating a further $2^{N_{CV}}$ additional nodes, and so on. This is illustrated in Figure 6.2 for two connected variables where the nodes for the first three iterations are numbered 1 through 12. Note that a maximum of 2^2 nodes are generated at each iteration and that at any given iteration K, all generated solution nodes share the same parent node. Such a structure has certain advantageous features, namely:

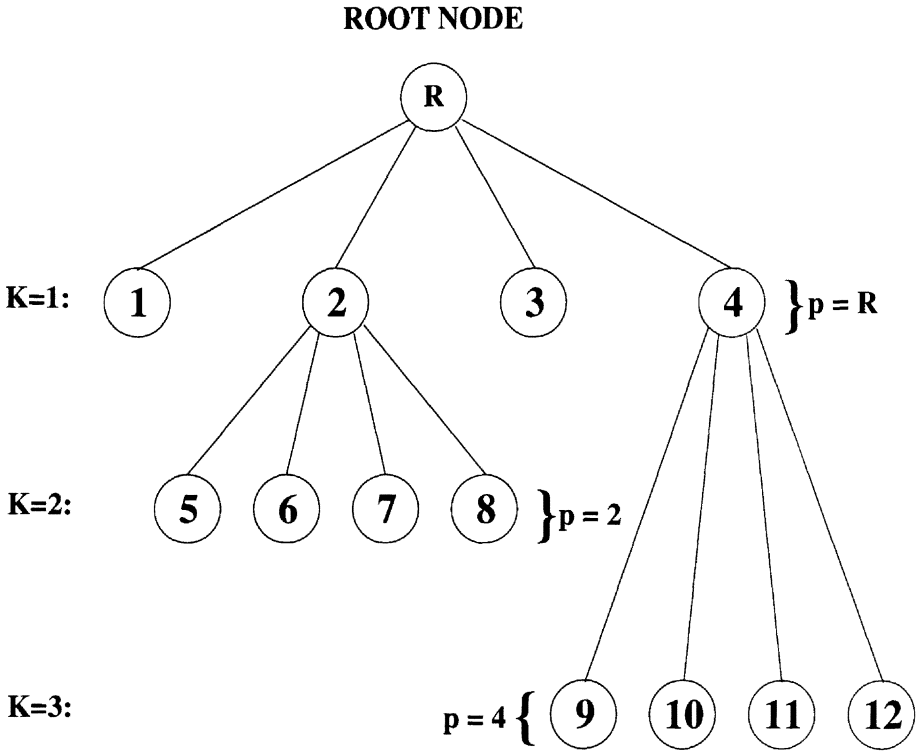


Figure 6.2. Tree structure for solution storage

(i) The *qualifying* constraints are a *single* set of box bounds on the variables, rather than sets of constraints generated from current and previous iterations.

(ii) Each n-rectangle is a refinement of its parent n-rectangle so that the regional bounds for a given node are supplied by the box bounds of its parent node. This has the important implication that any given Lagrangian will be valid in any future n-rectangles that it spawns.

(iii) Because of (ii), retrieving previous Lagrangians, labeled K_P, for use in the current relaxed dual subproblem is especially simple. A backward depth-first traversal through the solution tree from the current node to the root node,

extracting the relevant information required to construct the Lagrangian at each node along the path, generates the set of valid Lagrangians, denoted $PL(K_P)$. This set of Lagrange functions is included in *each* relaxed dual subproblem.

Remark 6.2.5 Point (iii) above brings the extremely important computational advantage that each relaxed dual subproblem contains relatively few Lagrangians from previous iterations. This is because a Lagrangian is not included for *all* previous iterations, but only for those whose nodes define the current node as a subdomain in the $\boldsymbol{y}$ variable space. This is shown in Figure 6.2 where only one Lagrangian from previous iterations is used at both the second *and* third iteration. Thus, each relaxed dual subproblem can be both generated and solved efficiently. The manner in which the x variable bounds are obtained and set on the basis of the qualifying constraints will now be described.

6.2.5.3 Evaluating x variable bounds. It is necessary to establish upper and lower bounds on the $\boldsymbol{x}$ variables within any given n-rectangle defined by $B\{\mathcal{L}^B, \mathcal{U}^B\}$. Recall that $\{\Psi_i^k\}$ are defined as linear fractionals. Any linear fractional is a pseudolinear function, that is, it is pseudoconvex *and* pseudoconcave. Thus, there is one local minimum and one local maximum that satisfy the KKT optimality conditions, and these will be unique global extrema. By examining the KKT conditions $\forall\, i \in C$, $\forall\, k \in P_L$ for the following problem:

$$\min \quad \Psi_i^k \quad s.t. \quad \mathcal{L}_{n_j^k}^B \leq n_j^k \leq \mathcal{U}_{n_j^k}^B \qquad \forall\, j \in C \tag{6.34}$$

the global minimum value for each Ψ_i^k in $B\{\mathcal{L}^B, \mathcal{U}^B\}$ can be evaluated and is labeled $\mathcal{L}_{\Psi_i^k}$. $-\Psi_i^k$ is minimized subject to the same constraints to obtain the corresponding maximum, $\mathcal{U}_{\Psi_i^k}$. McDonald and Floudas (1995a) showed that the globally valid lower and upper bounds for Ψ_i^k (and hence $\hat{\Psi}_{ij}^k$) within the n-rectangle defined by $B\{\mathcal{L}^B, \mathcal{U}^B\}$ are given as:

$$\mathcal{L}_{\Psi_i^k} = \frac{\mathcal{L}_{n_i^k}^B}{\mathcal{L}_{n_i^k}^B + \sum\limits_{j \neq i} \mathcal{G}_{ji}\, \mathcal{U}_{n_j^k}^B} \qquad \text{and} \qquad \mathcal{U}_{\Psi_i^k} = \frac{\mathcal{U}_{n_i^k}^B}{\mathcal{U}_{n_i^k}^B + \sum\limits_{j \neq i} \mathcal{G}_{ji}\, \mathcal{L}_{n_j^k}^B}. \tag{6.35}$$

6.2.5.4 Setting the bounds on the x variables. The final expression for the Lagrange function collects the terms in $\boldsymbol{n}$ to yield:

$$\begin{aligned} L(\hat{\boldsymbol{\Psi}}, \boldsymbol{n}, \bar{\boldsymbol{\lambda}}) \;&=\; \sum_{i \in C} \sum_{k \in P_L} \left\{ [n_i^k - \bar{n}_i^k] \cdot \sum_{j \in C} \hat{\Psi}_{ji}^k \left[\mathcal{G}_{ij} \left\{ \tau_{ij} + \bar{\lambda}_{\Psi_i^k} \right\} \right] \right\} \\ &+\; \sum_{k \in P} \mathcal{C}^k - \sum_{i \in C} \sum_{k \in P_L} n_i^k \bar{\lambda}_{\Psi_i^k}. \end{aligned} \tag{6.36}$$

For some $B_l \in CB$, it is required to set the bounds on $\hat{\Psi}_{ji}^k$ for the current iteration, K, and the set of previous iterations, K_P. Two quantities decide

whether the $\boldsymbol{x}$ variables are set at their lower or upper bounds. The first of these is the sign of the term $n_i^k - \bar{n}_i^k$ for current and previous iterations, and the second is the parameter $\bar{s}_{ji}^k$, used to determine the sign of the term that post–multiplies the x variables in Eqn. (6.36) as follows:

$$\bar{s}_{ji}^k = \begin{cases} +1 & \text{if} \quad \mathcal{G}_{ij}\left[\, \tau_{ij} + \bar{\lambda}_{\Psi_i^k} \,\right] \geq 0 \\ -1 & \text{otherwise.} \end{cases}$$

If the combined sign of these terms is positive, then the $\boldsymbol{x}$ variable is set to its *lower* bound, and if it is negative, then the $\boldsymbol{x}$ variable is set to its *upper* bound so that the corresponding terms in the Lagrange functions are validly underestimated. For the current iteration, K, $s_{ik}^{B_l}$ clearly determines the sign of $n_i^k - \bar{n}_i^k$, while for the set of previous iterations, $PL(K_P)$, the sign of the quantity $(\bar{n}_i^k)^K - (\bar{n}_i^k)^{K_P}$ properly determines the sign of $n_i^k - (\bar{n}_i^k)^{K_P}$ in the current region. In summary, the following set of steps performed for each $\{i, j, k\} \in C \times C \times P_L$ provide the correct bounds on the x variables:

<u>Current Iteration, K:</u>

$$\begin{aligned} \text{If} \quad (\bar{s}_{ji}^k)^K \cdot s_{ik}^{B_l} = +1 \quad \text{then} \quad (\hat{\Psi}_{ji}^k)^{B_l^K} &= \mathcal{L}_{\Psi_j^k} \\ \text{If} \quad (\bar{s}_{ji}^k)^K \cdot s_{ik}^{B_l} = -1 \quad \text{then} \quad (\hat{\Psi}_{ji}^k)^{B_l^K} &= \mathcal{U}_{\Psi_j^k} \end{aligned}$$

<u>Previous Iterations, K_P:</u>

$$\begin{aligned} \text{If} \quad (\bar{s}_{ji}^k)^{K_P} \cdot \left[\, (\bar{n}_i^k)^K - (\bar{n}_i^k)^{K_P} \,\right] = +1 \quad \text{then} \quad (\hat{\Psi}_{ji}^k)^{B_l^{K_P}} &= \mathcal{L}_{\Psi_j^k} \\ \text{If} \quad (\bar{s}_{ji}^k)^{K_P} \cdot \left[\, (\bar{n}_i^k)^K - (\bar{n}_i^k)^{K_P} \,\right] = -1 \quad \text{then} \quad (\hat{\Psi}_{ji}^k)^{B_l^{K_P}} &= \mathcal{U}_{\Psi_j^k} \end{aligned}$$

Illustration – *conclusion*: The new set of x variables is defined as:

$$\hat{\Psi}_{11} = \hat{\Psi}_{12} = \Psi_1 \quad \text{and} \quad \hat{\Psi}_{21} = \hat{\Psi}_{22} = \Psi_2$$

Collecting the variables in n_i together gives the final Lagrange function as:

$$\begin{aligned} L^4(\,\boldsymbol{\Psi}, \boldsymbol{n}, \bar{\boldsymbol{\lambda}}) &= C - n_1\bar{\lambda}_1 - n_2\bar{\lambda}_2 \\ &+ \{\, n_1 - \bar{n}_1 \,\}\left\{\, \hat{\Psi}_{11} \cdot \bar{\lambda}_1 + \hat{\Psi}_{21} \cdot \mathcal{G}_{12}\left[\, \tau_{12} + \bar{\lambda}_2 \,\right] \,\right\} \\ &+ \{\, n_2 - \bar{n}_2 \,\}\left\{\, \hat{\Psi}_{12} \cdot \mathcal{G}_{21}\left[\, \tau_{21} + \bar{\lambda}_1 \,\right] + \hat{\Psi}_{22} \cdot \bar{\lambda}_2 \,\right\}. \end{aligned}$$

Each term in n_i multiplies multiple a summation of terms involving the $\boldsymbol{x}$ variables. The bounds on these $\boldsymbol{x}$ variables are set in the manner just described in this section.

6.2.6 Global Optimization Algorithm for the NRTL Model

In what follows, k_S represents any node of the solution tree and the set of nodes is labeled N_{k_S}. k_t is a temporary node used in the generation of the set of previous Lagrange functions, $PL(K_P)$. $\mathcal{S}_C$ represents the current node under consideration at any given iteration and is obviously a leaf node. The subscript C denotes any parameter associated with the current iteration. The parent of a node is simply indicated by p. The set I_{k_S} represents the iteration number at which the particular node k_S is generated. The complete algorithm for the NRTL equilibrium model is now given.

STEP 0: Initialization

Select $\bar{\boldsymbol{n}}^0$ and ε. Set $K = 0$, $\mathcal{S}_C = R$, $k_S = \emptyset$, $P^U = +\infty$, $M^L = -\infty$.
Initialize $R_C\{\mathcal{L}^R, \mathcal{U}^R\}$ (viz. $\mathcal{L}^R_{n_i^k} = 0$; $\mathcal{U}^R_{n_i^k} = n^T$).

STEP 1: Primal Problem

Evaluate (G-P) to give $\hat{G}(\bar{\boldsymbol{n}}^K)$. Store $\bar{\boldsymbol{n}}^K, \bar{\boldsymbol{\lambda}}^K, (\bar{s}^k_{ji})^K$.
If $\hat{G}(\bar{\boldsymbol{n}}^K) < P^U$ solve (G-N) locally to give $\hat{G}^*$.
Update $P^U = \min\left[\, P^U \,,\, \hat{G}(\bar{\boldsymbol{n}}^K) \,,\, \hat{G}^* \,\right]$

STEP 2: Select previous Lagrangians

Set $PL(K_P) = \emptyset$, $k_t = S_C$.
while ($k_t \neq R$)
do
$K_P = I_{k_t}$
$PL(K_P) = PL(K_P) \cup K_P$
$k_t = p\,(k_t)$
end

STEP 3: The Relaxed Dual Phase

(1) Choose a combination of qualifying constraints, B_l from the set CB.
Use $s^{B_l}_{ik}$ and $R_C\{\mathcal{L}^R, \mathcal{U}^R\}$ to calculate $B\{\mathcal{L}^B, \mathcal{U}^B\}$ and $\{\mathcal{L}_{\Psi_i^k}, \mathcal{U}_{\Psi_i^k}\}$.
Set $\hat{\boldsymbol{\Psi}}^{B_l^K}$ and $\hat{\boldsymbol{\Psi}}^{B_l^{K_P}}$ and solve the following problem (G-RD) to give μ^*_B

and $\boldsymbol{n}^*$:

$$\left.\begin{array}{ll} \min & \mu_B \\ s.t. & \mu_B \geq L_C(\hat{\boldsymbol{\Psi}}^{B_l^K}, \boldsymbol{n}, \bar{\boldsymbol{\lambda}}^K) \\ & \mu_B \geq L(\hat{\boldsymbol{\Psi}}^{B_l^{K_P}}, \boldsymbol{n}, \bar{\boldsymbol{\lambda}}^{K_P}) \;\; \forall\, K_P \in PL(K_P) \\ & \mathcal{L}^B_{n_i^k} \leq \; n_i^k \; \leq \; \mathcal{U}^B_{n_i^k} \qquad \forall\; i \in C\;,\; k \in P_L \\ & \mathbf{0} = \mathbf{A} \cdot \boldsymbol{n} \; - \; \mathbf{b} \end{array}\right\} \text{(G-RD)}$$

where $L_C(\hat{\boldsymbol{\Psi}}^{B_l^K}, \boldsymbol{n}, \bar{\boldsymbol{\lambda}}^K)$ is given by Eqn. (6.36).

(i) If $\mu_B^* \geq P^U$, then fathom solution.

(ii) If $\mu_B^* < P^U$, then set $k_S = k_S + 1$, $p\,(k_S) = \mathcal{S}_C$, $I_{k_S} = K$, $\mu^{k_S} = \mu_B^*$, $\bar{\boldsymbol{n}}^{k_S} = \boldsymbol{n}^*$ and $R_{k_S}\{\mathcal{L}^R, \mathcal{U}^R\} = B\{\mathcal{L}^B, \mathcal{U}^B\}$.

(2) Choose another set of bounds B_l from CB and return to (1).

If there are no remaining unchosen B_l in CB, then proceed to Step 4.

STEP 4: Select mole vector for next iteration

$M^L = \underset{N_{k_S}}{\text{argmin}}\; \mu^{k_S}$; set $\mathcal{S}_C = k_S$, the associated node. Set $N_{k_S} = N_{k_S} \setminus \mathcal{S}_C$.

Set $\bar{\boldsymbol{n}}^{K+1} = \bar{\boldsymbol{n}}^{\mathcal{S}_C}$, $R_{K+1}\{\mathcal{L}^R, \mathcal{U}^R\} = R_{\mathcal{S}_C}\{\mathcal{L}^R, \mathcal{U}^R\}$.

STEP 5: Check for convergence

Check if $\left|\frac{P^U - M^L}{P^U}\right| \leq \epsilon_r$. If true, then STOP; otherwise set $K = K + 1$, and return to Step 1.

It has been shown how all the conditions required to guarantee ϵ-global convergence of the GOP algorithm are satisfied, (see the proof in Chapter 3 and in Floudas and Visweswaran (1990, 1993)).

Remark 6.2.6 The main computational effort lies in solving the relaxed dual subproblems. There is a very simple way to reduce the number of connected variables. The material balance constraints appear affinely in the relaxed dual formulation. The material balance matrix represented by A has rank r so that r mole number variables can be written in terms of the others. In other words r connected variables are eliminated. Thus, the number of connected variables

is now given as:

$$N_{CV} = \quad |C| \cdot |P| - \; r. \tag{6.37}$$

For the phase equilibrium problem $r = |C|$. For cases where only liquid phases are considered, the material balance constraints are used to eliminate these r variables from interacting with the $\boldsymbol{x}$ variable set. This is shown in the illustrative example in the next section. Another computational aid is the fact that at a given stage of the algorithm, if the current point matches a previous point for some (or all) $i \in C$, $k \in P_L$, then there is no need for a partitioning hyperplane in that dimension. This reduces the number of relaxed dual subproblems to be solved at that iteration.

6.3 COMPUTATIONAL STUDIES

Two examples are presented which demonstrate the GOP algorithm as it applies to the NRTL activity coefficient model. Both these problems have two postulated liquid phases, so that the Gibbs free energy of formation terms can be eliminated. The function to be minimized is then given as $\hat{G}_I = [G - \sum_i \Delta G_i^{L,f} n_i^T]/RT$, where G is defined by Eqn. (6.9). The first example illustrates the GOP approach, while the second example considers a difficult problem of three components for which most local optimization solvers converge to the erroneous trivial solution.

6.3.1 Illustrative Example: n-Butyl-Acetate and Water

This illustrative example was studied by Heidemann and Mandhane (1973) to demonstrate the potential complexities of the NRTL equation. It features two components, n-butyl-acetate (1) and water (2), at a temperature of 298K and a pressure of 1 atm. There are two possible liquid phases and they are modeled using the NRTL equation. Both phases share the same standard state, so that $\hat{G}_I$ as defined by Eqn. (6.15) supplies the Gibbs function to be minimized. The required binary parameters were obtained from Heidemann and Mandhane (1973):

$$\tau_{12} = 3.00498 \; , \; \tau_{21} = 4.69071 \quad \text{and} \quad \mathcal{G}_{12} = 0.30794 \; , \; \mathcal{G}_{21} = 0.15904.$$

The nonrandomness constant used to calculate $\mathcal{G}_{ij}$ is $\alpha_{12} = \alpha_{21} = 0.39196$. The initial mixture charge is equimolar ($n_i^T = 0.5 \; \forall \; i$) and no reaction occurs in the system.

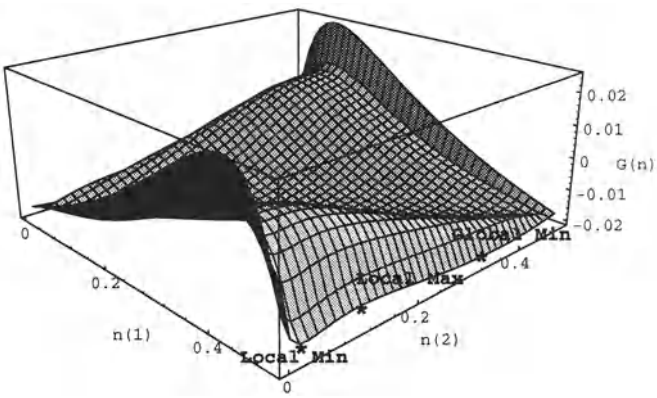


Figure 6.3. Gibbs energy surface for Illustrative Example

The explicit mathematical formulation is given as:

$$
\begin{aligned}
\min\ \hat{G}_I \quad &= \quad n_1^1 \ln n_1^1 + n_2^1 \ln n_2^1 - [n_1^1 + n_2^1] \ln[n_1^1 + n_2^1] \\
&+ \quad n_1^2 \ln n_1^2 + n_2^2 \ln n_2^2 - [n_1^2 + n_2^2] \ln[n_1^2 + n_2^2] \\
&+ \quad \mathcal{G}_{12}\tau_{12} n_1^1 \Psi_2^1 + \mathcal{G}_{21}\tau_{21} n_2^1 \Psi_1^1 + \mathcal{G}_{12}\tau_{12} n_1^2 \Psi_2^2 + \mathcal{G}_{21}\tau_{21} n_2^2 \Psi_1^2 \\
s.t. \quad & \quad \Psi_1^1\{n_1^1 + \mathcal{G}_{21} n_2^1\} = n_1^1 \\
& \quad \Psi_2^1\{\mathcal{G}_{12} n_1^1 + n_2^1\} = n_2^1 \\
& \quad \Psi_1^2\{n_1^2 + \mathcal{G}_{21} n_2^2\} = n_1^2 \\
& \quad \Psi_2^2\{\mathcal{G}_{12} n_1^2 + n_2^2\} = n_2^2 \\
& \quad n_1^1 + n_1^2 = 0.5 \\
& \quad n_2^1 + n_2^2 = 0.5 \\
& \quad 0 \le n_1^1, n_2^1, n_1^2, n_2^2 \le 0.5.
\end{aligned}
$$

It appears to be a simple example but there are multiple stationary points and local solutions. In fact, there is a local minimum *and* a local maximum, in addition to the global solution. There is also a line of trivial solutions that represents physically one phase behavior, but mathematically yields two phase solutions, that is, the mole fractions are the same in each distinct phase. These solutions are given in Table 6.1 where the superscript I represents the first liquid phase. The mole numbers for the second liquid phase, denoted by the

Solution for n-Butyl-Acetate (1) – Water (2) $T = 298$K, $P = 1.0$ atm				
Components	Feed (moles)	Liquid I (moles)	Liquid II (moles)	$\hat{G}_I^*$ (—)
$C_6H_{12}O_2$ (1)	0.50	0.00071	0.49929	-0.02020
H_2O (2)	0.50	0.15588	0.34412	(Global minimum)
$C_6H_{12}O_2$ (1)	0.50	0.00213	0.49787	-0.01961
H_2O (2)	0.50	0.46547	0.03453	(Local minimum)
$C_6H_{12}O_2$ (1)	0.50	0.00173	0.49827	-0.01730
H_2O (2)	0.50	0.37544	0.12456	(Local maximum)

Table 6.1. Solutions for Illustrative Example

superscript II, are obtained as $n_i^{II} = n_i^T - n_i^I \ \forall\, i$. The Gibbs free energy surface as a function of the mole numbers in liquid phase I is pictured in Figure 6.3, where the various local solutions are shown, with the trivial solutions lying along the line defined by $n_1^I - n_2^I = 0$. Lin (1988) employed a successive continuation method to solve the problem and trace all possible solution branches. In this manner, all the local and global extrema were obtained. However, there is no guarantee that a local solver will obtain the global solution. To illustrate this point, when (G-N) was solved using MINOS5.4 from 100 randomly selected starting points, the global solution was found in only 13 cases. The strong local minimum solution was found in 5 cases, and the trivial solution was obtained in the remaining 82 cases. This example therefore serves as an excellent demonstration tool for the proposed global optimization approach: it is challenging and it can be represented in three dimensional space.

In what follows the superscript I is dropped for the first liquid phase. The mass balance constraints are then given as:

$$n_1 + n_1^{II} = n_1^T \quad \text{and} \quad n_2 + n_2^{II} = n_2^T.$$

These can be used to reduce the number of connected variables to $N_{CV} = 2$ by substituting them into the terms involving n_i^{II} within the curly braces of the Lagrange function defined by Eqn. (6.36). Thus, a maximum of four relaxed dual problems must be solved at each iteration. The convex portion of the

objective function for the first liquid phase is defined as:

$$\mathcal{C}^1 = n_1 \ln \frac{n_1}{n_1 + n_2} + n_2 \ln \frac{n_2}{n_1 + n_2}$$

with $\mathcal{C}^{II}$ similarly defined for the second phase, then the explicit Lagrange function for use in the relaxed dual is as follows:

$$\begin{aligned} L(\boldsymbol{\Psi}, \boldsymbol{n}, \bar{\boldsymbol{\lambda}}) &= \mathcal{C} + \mathcal{C}^{II} - n_1\bar{\lambda}_1 - n_2\bar{\lambda}_2 - n_1^{II}\bar{\lambda}_1^{II} - n_2^{II}\bar{\lambda}_2^{II} \\ &+ \{n_1 - \bar{n}_1\} \cdot \left\{\hat{\Psi}_{11}\bar{\lambda}_1 + \hat{\Psi}_{21}\mathcal{G}_{12}\left[\tau_{12} + \bar{\lambda}_2\right]\right\} \\ &- \{n_1 - \bar{n}_1\} \cdot \left\{\hat{\Psi}_{11}^{II}\bar{\lambda}_1^{II}\hat{\Psi}_{21}^{II}\mathcal{G}_{12}\left[\tau_{12} + \bar{\lambda}_2^{II}\right]\right\} \\ &+ \{n_2 - \bar{n}_2\} \cdot \left\{\hat{\Psi}_{12}\mathcal{G}_{21}\left[\tau_{21} + \bar{\lambda}_1\right] + \hat{\Psi}_{22}\bar{\lambda}_2\right\} \\ &- \{n_2 - \bar{n}_2\} \cdot \left\{\hat{\Psi}_{12}^{II}\mathcal{G}_{21}\left[\tau_{21} + \bar{\lambda}_1^{II}\right] + \hat{\Psi}_{22}^{II}\bar{\lambda}_2^{II}\right\}. \end{aligned}$$

Notice that the terms on the first line are convex with the linear terms involving $\bar{\boldsymbol{\lambda}}$ arising from the inclusion of the right hand side of the bilinear equality constraints of (G-N) in the Lagrange function. The interaction between the $\boldsymbol{x}$ and the $\boldsymbol{y}$ variable sets on the next two lines is purely bilinear. The derivatives of the Lagrangian with respect to the $\boldsymbol{x}$ variables yield linear functionalities in the $\boldsymbol{y}$ variables allowing the $\boldsymbol{y}$ space to be partitioned in a very simple manner.

<u>INITIALIZATION</u>: Choose a center starting point: $\bar{n}_1 = n_1^T/2 = 0.25$, $\bar{n}_2 = n_2^T/2 = 0.25$.

<u>ITERATION 1</u>: The initial parent n-rectangle, labeled $R\{\mathcal{L}^R, \mathcal{U}^R\}$, is defined as the complete feasible region and the primal is solved (a function evaluation) to give a first upper bound of -0.01758. All the relevant information is supplied in Table 6.2. The parent n-rectangle is divided into 4 box regions, and a relaxed dual problem is solved in each. These 4 regions are shown in Figure 6.4 where the larger solid dots signify the current point of the iteration, while the smaller solid dots represent the locations of the solutions of the relaxed dual problems in the relevant subdomains. For Iteration 1, the qualifying constraints are also explicitly supplied in Figure 6.4.

Region 1: The box bounds are: $0.25 \leq n_1 \leq 0.5$, $0.25 \leq n_2 \leq 0.5$. The $\boldsymbol{x}$ variable bounds in this box region as calculated from Eqns. (6.35) are given in Table 6.3. Their actual levels in the Lagrangian are given in Table 6.4. As an illustration of how these bounds are set, consider the $\boldsymbol{x}$ variable $\hat{\Psi}_{21}$: the quantity $(n_1 - \bar{n}_1)$ is positive in Region 1. In addition:

$$\mathcal{G}_{12}\left[\tau_{12} + \bar{\lambda}_2\right] = 0.7075 \ > \ 0.$$

Therefore, the combined sign of this term will be positive. The $\boldsymbol{x}$ variable is set to its lower bound in this case, so that $\hat{\Psi}_{21} = \mathcal{L}_{\Psi_2}^B = 0.6189$, ensuring that the

	Parent region information						Multiplier levels			
Iter.	$\mathcal{L}^R_{n_1}$	$\bar{n}_1$	$\mathcal{U}^R_{n_1}$	$\mathcal{L}^R_{n_2}$	$\bar{n}_2$	$\mathcal{U}^R_{n_2}$	λ_1	λ_2	λ_1^{II}	λ_2^{II}
1	0	0.25	0.5	0	0.25	0.5	−0.6436	−0.7075	−0.6436	−0.7075
2	0.25	0.5	0.5	0.25	0.5	0.5	−0.6436	−0.7075	0	0
3	0	0.1028	0.25	0.25	0.3972	0.5	−1.7848	−0.2219	−0.1855	−1.6323
4	0.25	0.3796	0.5	0.25	0.3704	0.5	−0.6302	−0.7207	−0.6853	−0.6687

Table 6.2. Primal information for Illustrative Example

	Liquid phase I				Liquid phase II			
Region	$\mathcal{L}^B_{\Psi_1}$	$\mathcal{U}^B_{\Psi_1}$	$\mathcal{L}^B_{\Psi_2}$	$\mathcal{U}^B_{\Psi_2}$	$\mathcal{L}^B_{\Psi_1^{II}}$	$\mathcal{U}^B_{\Psi_1^{II}}$	$\mathcal{L}^B_{\Psi_2^{II}}$	$\mathcal{U}^B_{\Psi_2^{II}}$
1	0.7587	0.9263	0.6189	0.8666	0	1	0	1
2	0	0.8628	0.7646	1	0.8628	1	0	0.7646

Table 6.3. x variable bounds for Iteration 1

	Liquid Phase I				Liquid Phase II			
Region	$\hat{\Psi}_{11}$	$\hat{\Psi}_{12}$	$\hat{\Psi}_{21}$	$\hat{\Psi}_{22}$	$\hat{\Psi}^{II}_{11}$	$\hat{\Psi}^{II}_{12}$	$\hat{\Psi}^{II}_{21}$	$\hat{\Psi}^{II}_{22}$
1	0.9263	0.7587	0.6189	0.8666	0	1	1	0
2	0	0	1	1	1	1	0	0

Table 6.4. x variable levels for Iteration 1

lower bound times a positive quantity will always provide a valid lower bound in the box region for this term. By performing the same analysis for each of the x variables, it is ensured that the Gibbs surface is validly underestimated

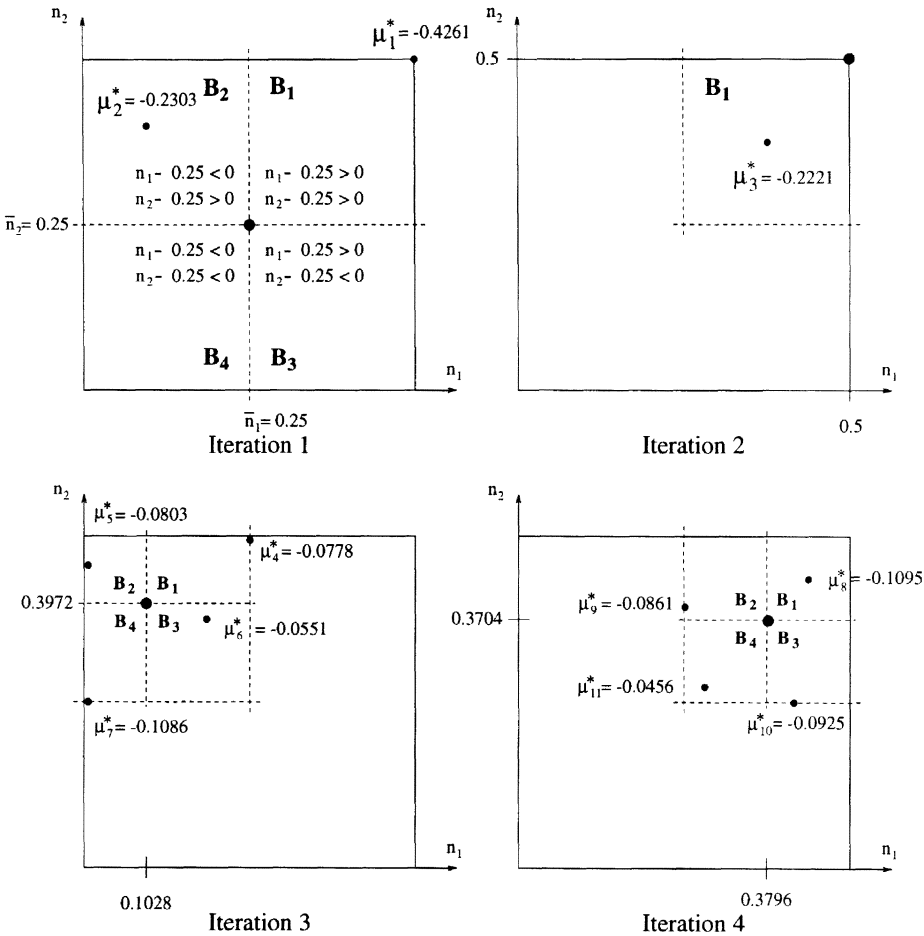


Figure 6.4. Box region and solution locations in the y variable space

in each box region of interest. The Gibbs surface in this region is shown in Figure 6.5 along with its underestimating Lagrangian. Notice that (i) the Lagrangian matches the Gibbs surface at the current point $\bar{n} = \{0.25, 0.25\}$, (ii) that it underestimates the Gibbs surface at all points, and (iii) that it is clearly convex. The solution of the relaxed dual lies in the bottom left hand corner of Figure 6.5 at a vertex of the box region, and is given as:

$$n_1^* = n_2^* = 0.5 \quad \text{with} \quad \mu_1^* = -0.4261.$$

Region 2: The box bounds are: $0 \leq n_1 \leq 0.25$, $0.25 \leq n_2 \leq 0.5$. The Gibbs energy surface and the corresponding Lagrangian are shown in Figure 6.6. In

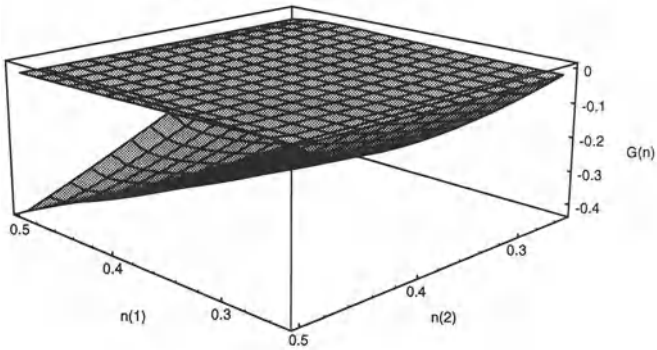


Figure 6.5. Gibbs surface and Lagrangian for Region 1 of Iteration 1

this case the solution lies in the interior of the box region and is:

$$n_1^* = 0.1028\ ,\ n_2^* = 0.3972 \quad \text{with} \quad \mu_2^* = -0.2303.$$

Region 3: The box bounds are: $0.25 \leq n_1 \leq 0.5$, $0 \leq n_2 \leq 0.25$. The solution is given as follows:

$$n_1^* = 0.3972\ ,\ n_2^* = 0.1028 \quad \text{with} \quad \mu^* = -0.2303.$$

Region 4: The box bounds are: $0 \leq n_1 \leq 0.25$, $0 \leq n_2 \leq 0.25$ with solution:

$$n_1^* = 0\ ,\ n_2^* = 0 \quad \text{with} \quad \mu^* = -0.4261.$$

Note that Regions 3 and 4 yield the same lower bounds as Regions 2 and 1 respectively and that the $\boldsymbol{y}$ variable levels in these regions are equal to $\{n_i^T - n_i^*\}$ where $\{n_i^*\}$ represents the appropriate solution from either Region 1 or 2. This is due to the fact that a starting point in the center of the feasible region was chosen so that the Lagrangian yields equivalent expressions for $\{n_i\}$ as for $\{n_i^T - n_i\}$, implying that Regions 3 and 4 may be discarded from future consideration. For other starting points this property will not hold.

ITERATION 2: The infimum of all obtained relaxed dual solutions supplies the parent region for this iteration as Region 1 of Iteration 1, with a lower bound of

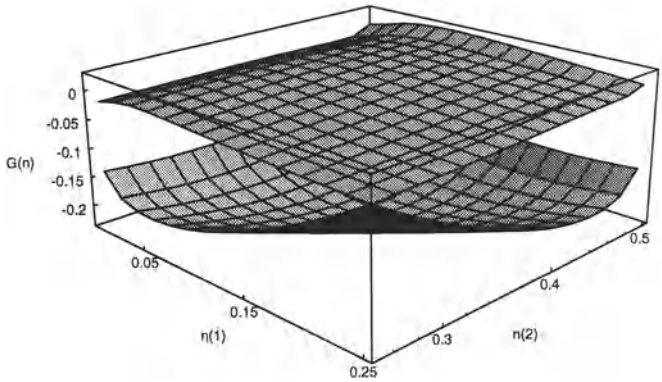


Figure 6.6. Gibbs surface and Lagrangian for Region 2 of Iteration 1

-0.4261. This solution is then deleted from the candidate set of lower bounds. All information in regard to the primal is given in Table 6.2. The current point is $\bar{n} = \{0.5, 0.5\}$ and thus the upper bound given by the primal does not change, that is, it also lies along the hyperplane of trivial solutions. Only one region need be considered in this case as the current point lies on a vertex of the parent region. For this reason, the box region is the same as it was for Region 1 of Iteration 1 as shown in Figure 6.4. Two Lagrange functions are required. For the Lagrange function to be generated from the first iteration, $(\bar{n}_i)^2 - (\bar{n}_i)^1 = 0.5 - 0.25 > 0$ for $i = 1, 2$. $n_i - (\bar{n}_i)^2 = n_i - 0.5 < 0$ for the current iteration. By way of illustration, for the Lagrange function of the first iteration, $(\hat{\Psi}_{11})^1 = \mathcal{U}_{\Psi_1} = 0.9263$ as $(\bar{\lambda}_1)^1 < 0$ and $n_i - 0.25 > 0$; while for the current iteration, $(\hat{\Psi}_{11})^2 = \mathcal{L}_{\Psi_2} = 0.7587$ as $(\bar{\lambda}_1)^2 < 0$ and $n_i - 0.5 < 0$. For both these cases, $(\bar{\lambda}_1)^K < 0$. The Gibbs surface and the supremum of the Lagrangian generated in Region 1 of Iteration 1 and the current one are plotted in Figure 6.7. The solution lies in the interior of the feasible region:

$$n_1^* = 0.3796\ ,\ n_2^* = 0.3704 \quad \text{with} \quad \mu_3^* = -0.2221.$$

This lower bound lies below the current best upper bound and is added to the set of relaxed dual solutions.

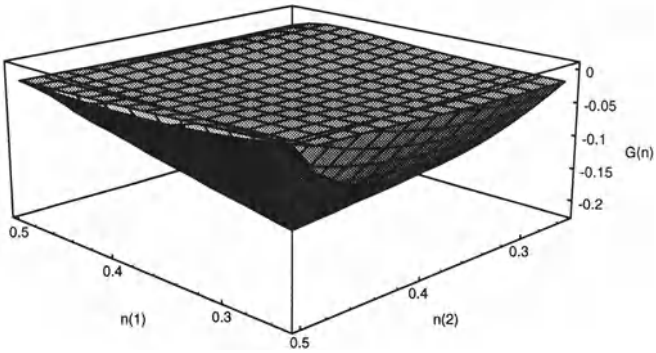


Figure 6.7. Gibbs surface and Lagrangian for Iteration 2

ITERATION 3: Region 2 of Iteration 1 contains the infimum of all remaining lower bounds as -0.2303 and is deleted from the set of candidate solutions. The new parent region is therefore defined as: $0 \leq n_1 \leq 0.25$, $0.25 \leq n_2 \leq 0.5$, with a current point $\bar{n} = \{0.1028, 0.3972\}$. This point yields a primal value of 0.00507 implying that the current best upper bound does not change. It is divided into four box regions as is illustrated in Figure 6.4. The box bounds for these regions along with the corresponding solutions are supplied in Table 6.5. The locations of these solutions in the $\boldsymbol{y}$ variable space can also be seen in Figure 6.4. There are two Lagrangians for each box region, one from Region 2 of Iteration 1 which is the same for all four box regions, and one from the current iteration. The supremum of both these Lagrangians is plotted for all four regions along with the Gibbs surfaces in Figure 6.8.

ITERATION 4: The infimum of all remaining lower bounds is obtained from Iteration 2 as -0.2221. This solution is then deleted from the set of relaxed dual lower bounds. The parent region for this iteration is defined as: $0.25 \leq n_1 \leq 0.5$, $0.25 \leq n_2 \leq 0.5$, with a current point $\bar{n} = \{0.3796, 0.3704\}$. This point yields a primal value of -0.01754 so the current best upper bound does not change. Four relaxed dual problems must be solved and each of these contains 3 Lagrangians, two constructed from Iterations 1 and 2 (common to all 4 subproblems), and a third which is based on the particular box region

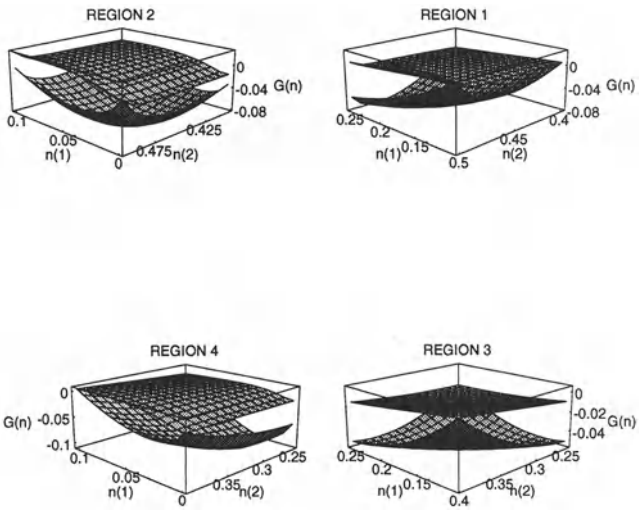


Figure 6.8. Gibbs surfaces and Lagrangians for Iteration 3

	Box bounds				Solutions		
Region	$\mathcal{L}^B_{n_1}$	$\mathcal{U}^B_{n_1}$	$\mathcal{L}^B_{n_2}$	$\mathcal{U}^B_{n_2}$	n_1^*	n_2^*	μ^*
1	0.1028	0.25	0.3972	0.5	0.25	0.4775	−0.0778
2	0	0.1028	0.3972	0.5	0.0315	0.4541	−0.0803
3	0.1028	0.25	0.25	0.3972	0.1828	0.3712	−0.0551
4	0	0.1028	0.25	0.3972	0.0226	0.25	−0.1086

Table 6.5. Relaxed dual information for Iteration 3

	Box bounds				Solutions		
Region	$\mathcal{L}^B_{n_1}$	$\mathcal{U}^B_{n_1}$	$\mathcal{L}^B_{n_2}$	$\mathcal{U}^B_{n_2}$	n_1^*	n_2^*	μ^*
1	0.3796	0.5	0.3704	0.5	0.4416	0.4333	-0.1095
2	0.25	0.3796	0.3704	0.5	0.2563	0.3989	-0.0861
3	0.3796	0.5	0.25	0.3704	0.4173	0.25	-0.0925
4	0.25	0.1028	0.25	0.3704	0.3144	0.3105	-0.0456

Table 6.6. Relaxed dual information for Iteration 4

under consideration. The box bounds are given in Table 6.6 together with the solutions of each of the relaxed dual problems. The 4 box regions and the location of these solutions is shown in Figure 6.4.

The algorithm proceeds in this manner until the infimum of all the remaining lower bounds is within $\epsilon_r = 0.0005$ of the best upper bound. For this particular problem and starting point, convergence occurs after 107 iterations. A local solver will *not* converge to the global solution from the initial point considered above. The progress of the upper and lower bounds is shown in Table 6.7. The total CPU time required was 1.23 sec, and a total of 359 relaxed dual subproblems were solved. 57% of these solutions were fathomed.

6.3.2 Example 2: Equilibrium for n-Propanol – n-Butanol – Water

This system was one of two studied by Block and Hegner (1976) in their modeling of three phase distillation towers. n-Butanol and water form the only partially miscible binary pair (i.e., it is a Type I system) with a relatively small domain of immiscibility. The binary parameters as obtained for use in the NRTL equation are supplied in Table 6.8. $\hat{G}_{II}$ supplies the objective function to be minimized, and $N_{CV} = 3$. Block and Hegner (1976) conducted the liquid phase splitting computations independently of the vapor phase that is, the parameters have no dependence on temperature. It is therefore meaningless to consider a vapor phase for this example. Walraven and van Rompay (1988) subsequently used this problem in order to test their phase splitting algorithm for a number of different feed charges.

Iteration	M^L	$P(\bar{\mathbf{n}})$
1	-0.42615	-0.01758
2	-0.23027	0.00507
3	-0.22209	-0.01754
4	-0.10955	-0.01752
47	-0.02501	-0.01980
73	-0.02169	-0.01988
86	-0.02082	-0.02002
90	-0.02048	-0.02018
92	-0.02039	-0.02019
107	-0.02020	-0.02020

Table 6.7. Progress of bounds for Illustrative Example

n-Propanol (1) – n-Butanol (2) – Water (3): τ_{ij} and α_{ij} dimensionless					
Components ij	i	j	τ_{ij}	τ_{ji}	$\alpha_{ij} = \alpha_{ji}$
C_3H_8O – $C_4H_{10}O$	1	2	−0.61259	0.71640	0.30
C_3H_8O – H_2O	1	3	−0.07149	2.7425	0.30
$C_4H_{10}O$ – H_2O	2	3	0.90047	3.51307	0.48

Table 6.8. Binary data for Example 2

Solutions for n-Propanol (1) – n-Butanol (2) – Water (3) at T, P = 1 atm							
Component	Feed (moles)	Liquid I (moles)	Liquid II (moles)	$\hat{G}^*_{II}$ (—)	cpu (sec)	N_I (—)	N_F (%)
C_3H_8O (1)	0.040	0.0049	0.0351	-1.24112	6.27	313	67
$C_4H_{10}O$ (2)	0.160	0.0095	0.1505				
H_2O (3)	0.800	0.4153	0.3847				
C_3H_8O (1)	0.148	0.1280	0.0200	-1.1919716	29.95	1490	27
$C_4H_{10}O$ (2)	0.052	0.0456	0.0064				
H_2O (3)	0.800	0.6549	0.1451				

Table 6.9. Global solutions for Example 2

The explicit formulation is :

$$\begin{aligned}
\min \hat{G}_I &= n_1^1 \ln n_1^1 + n_2^1 \ln n_2^1 + n_3^1 \ln n_3^1 - [n_1^1 + n_2^1 + n_3^1] \ln[n_1^1 + n_2^1 + n_3^1] \\
&+ n_1^2 \ln n_1^2 + n_2^2 \ln n_2^2 + n_3^2 \ln n_3^2 - [n_1^2 + n_2^2 + n_3^2] \ln[n_1^2 + n_2^2 + n_3^2] \\
&+ n_1^1 \left[\mathcal{G}_{12}\tau_{12}\Psi_2^1 + \mathcal{G}_{13}\tau_{13}\Psi_3^1\right] + n_2^1 \left[\mathcal{G}_{21}\tau_{21}\Psi_1^1 + \mathcal{G}_{23}\tau_{23}\Psi_3^1\right] \\
&+ n_3^1 \left[\mathcal{G}_{31}\tau_{31}\Psi_1^1 + \mathcal{G}_{32}\tau_{32}\Psi_2^1\right] \\
&+ n_1^2 \left[\mathcal{G}_{12}\tau_{12}\Psi_2^2 + \mathcal{G}_{13}\tau_{13}\Psi_3^2\right] + n_2^2 \left[\mathcal{G}_{21}\tau_{21}\Psi_1^2 + \mathcal{G}_{23}\tau_{23}\Psi_3^2\right] \\
&+ n_3^2 \left[\mathcal{G}_{31}\tau_{31}\Psi_1^2 + \mathcal{G}_{32}\tau_{32}\Psi_2^2\right]
\end{aligned}$$

s.t.

$$\begin{aligned}
\Psi_1^1\{n_1^1 + \mathcal{G}_{21}n_2^1 + \mathcal{G}_{31}n_3^1\} &= n_1^1 & \Psi_1^2\{n_1^2 + \mathcal{G}_{21}n_2^2 + \mathcal{G}_{31}n_3^2\} &= n_1^2 \\
\Psi_2^1\{\mathcal{G}_{12}n_1^1 + n_2^1 + \mathcal{G}_{32}n_3^1\} &= n_2^1 & \Psi_2^2\{\mathcal{G}_{12}n_1^2 + n_2^2 + \mathcal{G}_{32}n_3^2\} &= n_2^2 \\
\Psi_3^1\{\mathcal{G}_{13}n_1^1 + \mathcal{G}_{23}n_2^1 + n_3^1\} &= n_3^1 & \Psi_3^2\{\mathcal{G}_{13}n_1^2 + \mathcal{G}_{23}n_2^2 + n_3^2\} &= n_3^2
\end{aligned}$$

$$\begin{aligned}
n_1^1 + n_1^2 &= n_1^T & 0 \le n_1^1, n_1^2 &\le n_1^T \\
n_2^1 + n_2^2 &= n_2^T & 0 \le n_2^1, n_2^2 &\le n_2^T \\
n_3^1 + n_3^2 &= n_3^T & 0 \le n_3^1, n_3^2 &\le n_3^T.
\end{aligned}$$

Two source feeds from the work of Walraven and van Rompay (1988) were examined, and are shown in Table 6.9. The first of these lies well within the immiscibility region, $\{n_i^T\} = \{0.04, 0.16, 0.80\}$, and therefore causes little problem for a local solver. However, the second considered source feed of $\{n_i^T\} = \{0.148, 0.052, 0.800\}$, lies close to the plait point, an area in which it is notoriously difficult to obtain the correct equilibrium solution. The trivial

solution objective function value is -1.1919705, while the two phase global solution has an objective function value of -1.1919716, a difference of only $1.1 \cdot 10^{-6}$! Solving (G-N) using MINOS 5.4 succeeded in obtaining the global solution from only 8 out of 100 random starting points. Nonetheless, the GOP algorithm generated the global solution for this very difficult problem when supplied with a trivial solution starting point. This clearly demonstrates the effectiveness of the algorithm in generating global solutions for extremely challenging problems.

The equilibrium solutions for the two sets of conditions considered here are given in Table 6.9, along with some selected computational results. N_I is the number of iterations required to converge to the global solution, and N_F is the percentage of solutions that are fathomed. The difficulty of the problem when the source charge is close to the plait point is evident in the increased computational effort required to obtain the equilibrium solution for this case.

6.4 TANGENT PLANE STABILITY CRITERION

In this section, the tangent plane stability criterion will be introduced in preparation for the global optimization approaches. The Gibbs tangent plane criterion is an important tool in determining the quality of obtained solutions to the phase and chemical equilibrium problem. The ability to determine if a postulated solution is thermodynamically stable with respect to perturbations in any or all of the phases is very useful in the search for the true equilibrium solution.

6.4.1 Motivation

Before discussing and deriving the tangent plane criterion, the motivation for its use is briefly outlined. This arises from consideration of the Gibbs free energy surface and the manner in which the tangents to this surface can describe the nature of equilibrium. Consider a binary system that is characterized by a single thermodynamic model, which is unspecified as yet, and it is assumed to be first order continuous. Then a plot such as the one shown in Figure 6.9 is obtained where the molar Gibbs free energy curve, $g(\boldsymbol{x})$, is plotted versus the mol fraction, x, of one of the components.

The mole fraction of the second component is given by $1 - x$. Suppose that a two–phase solution corresponding to a *local* (not global) minimum of the Gibbs free energy function has the mole fraction of component one in phase one denoted by z^1 and in phase two by z^2, with associated chemical potentials $\mu^0(z^1)$ and $\mu^0(z^2)$. Note that the superscripts refer to phases, not components. This local solution satisfies the necessary first order condition of equal chemical potentials, so that $\mu^0(z^1) = \mu^0(z^2)$. If the tangents to the Gibbs surface are drawn at z^1 and z^2, then these correspond to the tangent plane to the Gibbs free energy surface, denoted $\mathcal{T}(z)$. This implies that points of common tangency reveal multiple phases. Note that the tangent line associated with this local solution lies above the Gibbs surface in a small region around $y \approx 0.6$, indicating thermodynamic instability. However, the situation for the solution

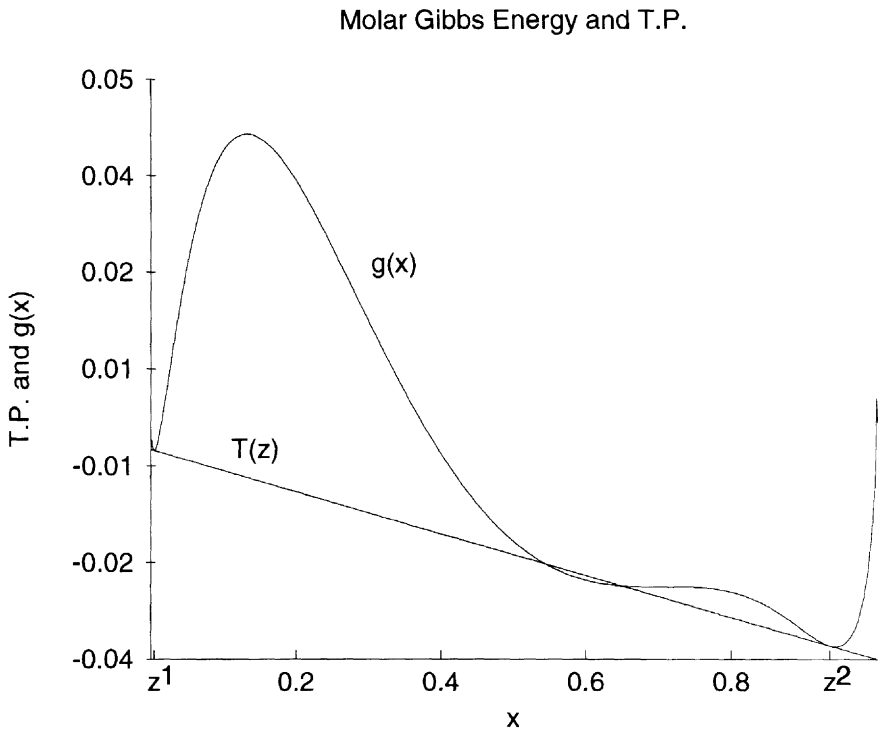


Figure 6.9. Illustration of stability for non-equilibrium case

corresponding to a global minimum is pictured in Figure 6.10 obtained for a different z^1 and z^2 than before. In this case, the common tangent lies completely below the Gibbs surface. For any mole fraction x that falls between z^1 and z^2, the hypothetical single liquid phase is unstable and will split into two phases with the compositions in each phase given by z^1 and z^2 respectively. The key characteristic is that if a tangent plane that lies completely below the Gibbs free energy surface does exist, then the equilibrium solution corresponding to the global minimum of the Gibbs free energy is given by the compositions at the points of tangency.

However, in the general multi–component, multi–phase problem there is no rigorous approach to determine where the tangent plane touches the Gibbs free energy surface so that it always lies below it. Rather, a point is picked on the composition space, which corresponds to some solution that satisfies the condition of equipotentials, and the tangent plane arising from this solution is checked to ensure that it lies below the Gibbs free energy surface for all feasible compositions.

We now show why negative tangent plane distances reveal instability. Assume that the system in question is at constant temperature and pressure. Then

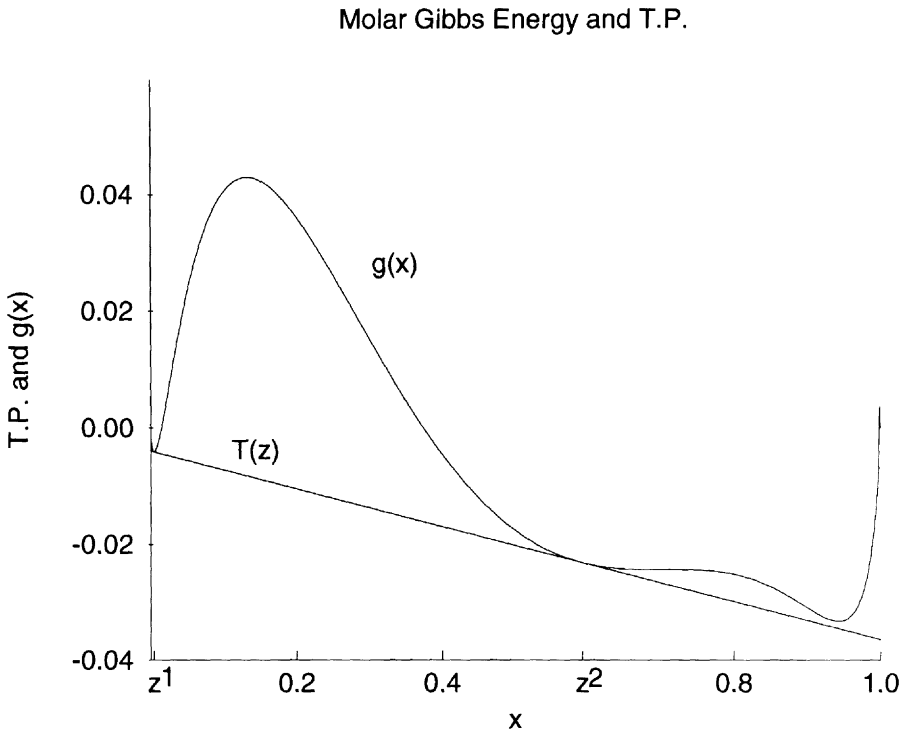


Figure 6.10. Illustration of stability for equilibrium case

the Gibbs free energy function is the thermodynamic quantity of interest. If the index set C represents the components i so that $C \equiv \{i\}$, then consider a single phase with mole fractions z_i and total mols n^0. An infinitesimal number of moles ϵ is split from the original phase to form two new phases, with the new phase having mol fractions x_i, and the original phase now containing $\mathrm{n}^0 - \epsilon$ moles. The change in the Gibbs free energy arising from this change is given as follows:

$$\begin{aligned}
\Delta G &= G(\epsilon) + G(\mathrm{n}^0 - \epsilon) - G(\mathrm{n}^0) \\
&= \sum_{i \in C} \epsilon\, x_i \mu_i(\boldsymbol{x}) + G(\mathrm{n}^0) - \sum_{i \in C} \epsilon\, x_i \left.\frac{\partial G}{\partial \epsilon x_i}\right|_{\boldsymbol{z}} - G(\mathrm{n}^0) \\
&= \sum_{i \in C} \epsilon\, x_i \left\{\mu_i(\boldsymbol{x}) - \mu_i^0(\boldsymbol{z})\right\}.
\end{aligned}$$

If this energy change is positive for all feasible values of $\boldsymbol{x}$, then the Gibbs free energy of the postulated phase cannot be reduced by adding, deleting or changing the relative amounts of the various components in the postulated solution, so that it is seen to be stable. On the other hand, if the energy change

is negative, then another solution must be sought which will have a lower value of the Gibbs free energy function.

6.4.2 Formulation

Baker *et al.* (1982) and Smith *et al.* (1993) provide extensive proofs that a necessary and sufficient condition for a postulated solution to be an equilibrium one is that the tangent plane distance function, denoted $\mathcal{F}(\boldsymbol{x})$ be nonnegative *for all possible phases represented in the system*, with $\mathcal{F}(\boldsymbol{x})$ defined as follows:

$$\mathcal{F}(\boldsymbol{x}) = \sum_{i \in C} x_i \left\{ \mu_i(\boldsymbol{x}) - \mu_i^0(\boldsymbol{z}) \right\} \tag{6.38}$$

where $\mu_i(\boldsymbol{x})$ and $\mu_i^0(\boldsymbol{z})$ represent the chemical potentials calculated at $\boldsymbol{x}$ and $\boldsymbol{z}$ respectively. The significance of the tangent plane criterion has been illustrated in Figures 6.9 and 6.10. The tangent plane distance function is defined as the distance between the Gibbs surface and the tangent plane constructed to this surface at z. This is summarized as follows:

- $g(\boldsymbol{x}) = \sum_{i \in C} x_i \mu_i(\boldsymbol{x})$ Molar Gibbs surface (for a single phase) .
- $\mathcal{T}(\boldsymbol{x}) = \sum_{i \in C} x_i \mu_i^0(\boldsymbol{z})$ Tangent plane to this surface at z .
- $\mathcal{F}(\boldsymbol{x}) = g(\boldsymbol{x}) - \mathcal{T}(\boldsymbol{x})$ Tangent plane distance function .

Clearly, if the tangent plane lies completely below the Gibbs surface as in Figure 6.10, then the postulated solution corresponds to a global minimum of the Gibbs free energy.

The tangent plane distance function is explicitly minimized. If a nonnegative global optimum solution is obtained, then the stability of the postulated solution is guaranteed. The optimization formulation for the phase stability problem, (S), is then given as:

$$\left.\begin{array}{ll} \min & \mathcal{F}(\boldsymbol{x}) = g(\boldsymbol{x}) - \sum_{i \in C} x_i \mu_i^0(\boldsymbol{z}) \\ s.t. & \sum_{i \in C} x_i = 1 \\ & 0 \leq x_i \leq 1. \end{array}\right\} \text{(S)}$$

Difficulties arise in obtaining global optimum solutions for (S) due to the complex and nonlinear nature of the models available to characterize $g(\boldsymbol{x})$ for nonideal phases.

Remark 6.4.1 Previous approaches have concentrated on obtaining the stationary points of the tangent plane distance function, which reduce to solving

the set of nonlinear equations:

$$\mu_i(\boldsymbol{x}) - \mu_i^0(\boldsymbol{z}) = K \tag{6.39}$$

subject to the feasibility constraints on $\boldsymbol{x}$. If any of the solutions to this set of equations is negative, then the postulated solution is unstable and an improved solution with a lower Gibbs free energy must be sought. If all stationary points are found to be nonnegative, then it is assumed that the postulated solution is stable. However, there are a number of difficulties associated with this approach. Firstly, no guarantee of obtaining all stationary points can be provided so that even if no negative solutions are obtained, the postulated configuration may be unstable. Also, the solutions obtained from the stability problem are then used to initiate the search for a solution with a lower Gibbs free energy. However, there is no way of telling which stationary points correspond to the global solution, so these guesses may in fact lead to local optima, or even infeasible solutions. If homotopy continuation methods are employed in an attempt to locate all stationary points, no theoretical guarantee of identifying them can be provided, as the phase stability problem contains logarithmic terms. Such guarantees can only be made for polynomial systems of equations.

6.5 APPLICATION TO THE NRTL EQUATION

In this section, a detailed derivation of the relaxed dual problem is presented. The presentation is based on the work of McDonald and Floudas (1995b). The liquid phase mole fractions are denoted by $\boldsymbol{y} \equiv \{y_i\}$ rather than $\boldsymbol{x}$, to avoid confusion with the $\boldsymbol{x}$ variables of the GOP.

6.5.1 Problem Formulation

The tangent plane distance function can be written in terms of the liquid phase mole fraction $\boldsymbol{y}$ as:

$$\begin{aligned}\mathcal{F}(\boldsymbol{y}) &= g(\boldsymbol{y}) - \sum_{i\in C} y_i \mu_i^0(\boldsymbol{z}) \\ &= \sum_{i\in C} y_i \left\{ \Delta G_i^f + \ln y_i + \frac{\sum_{j\in C} \tau_{ij}\mathcal{G}_{ij} y_j}{\sum_{l\in C} \mathcal{G}_{lj} y_l} - \mu_i^0(\boldsymbol{z}) \right\}.\end{aligned} \tag{6.40}$$

By finding the global minimum of $\mathcal{F}(\boldsymbol{y})$ over the convex constraint set of the phase stability problem (S), it is possible to determine whether the postulated solution represented by $\boldsymbol{z}$ is stable with respect to perturbations. If the global minimum is negative, then a solution with a lower Gibbs free energy must be sought by some other means. If a nonnegative value is obtained for the global minimum of $\mathcal{F}(\boldsymbol{y})$, then the solution is stable, and the postulated solution describes the equilibrium state.

To demonstrate the structure of the stability problem, it is worthwhile to collect the convex terms of the tangent plane distance function together, labeled

$\mathcal{C}(\boldsymbol{y})$, as follows:

$$\mathcal{C}(\boldsymbol{y}) = \sum_{i \in C} y_i \left\{ \Delta G_i^f + \ln y_i - \mu_i^0(\boldsymbol{z}) \right\} \tag{6.41}$$

where the linear terms arising from the slopes of the Gibbs free energy taken at $\boldsymbol{z}$ can obviously be included in the convex expression. This implies that the objective function can now be written as:

$$\mathcal{F}(\boldsymbol{y}) = \mathcal{C}(\boldsymbol{y}) + \sum_{i \in C} y_i \frac{\sum_{j \in C} \tau_{ij} \mathcal{G}_{ij} y_j}{\sum_{l \in C} \mathcal{G}_{lj} y_l}. \tag{6.42}$$

The nonconvexities in Eqn. (6.42) arise solely due to the excess Gibbs free energy term, and consist of a summation of terms that each involves a linear term times a linear fractional term. This is a nonconvex bilinear fractional term, so that if a local optimization technique is used to obtain the minimum of $\mathcal{F}(\boldsymbol{y})$, a *local* solution can at best be obtained.

Transformations and Partitioning

By introducing new variables labeled $\mathcal{X}_i$, it is possible to change the nature of the nonconvexities of Eqn. (6.42) so that the GOP algorithm (Floudas and Visweswaran, 1990, 1993) can be applied to the phase stability problem:

$$\mathcal{X}_i = \frac{y_i}{\sum_{j \in C} \mathcal{G}_{ji} y_j} \quad \forall\, i \in C. \tag{6.43}$$

Substituting these variables into Eqn. (6.42) yields the following reformulated stability problem for the NRTL activity coefficient equation (S-N) given below:

$$\left.\begin{array}{ll}
\min & \mathcal{F}(\boldsymbol{y}) = \mathcal{C}(\boldsymbol{y}) + \sum_{i \in C} y_i \cdot \sum_{j \in C} \mathcal{G}_{ij} \tau_{ij} \mathcal{X}_j \\
s.t. & \mathcal{X}_i \cdot \sum_{j \in C} \mathcal{G}_{ji} y_j = y_i \quad \forall\, i \in C \\
 & \sum_{i \in C} y_i = 1 \\
\text{with} & Y \equiv \{ y_i : 0 \leq y_i \leq 1 \quad \forall\, i \in C \}.
\end{array}\right\} \text{(S-N)}$$

Note that the constraints of Eqn. (6.43) have been rewritten so that they are of a bilinear form, rather than linear fractional.

Remark 6.5.1 (S-N) reveals that there is a natural partition of the variables into two distinct subsets to ensure that a convex subproblem remains if one of

these subsets is held fixed. If the variables are partitioned so that the original mole number variables belong to the set of $\boldsymbol{y}$ variables (i.e., $\boldsymbol{y} \leftarrow \{y_i\}$), and the new introduced variables belong to the set of $\boldsymbol{x}$ variables (i.e., $\boldsymbol{x} \leftarrow \{\mathcal{X}_i\}$). Then *Conditions* (A) required by the GOP algorithm (Floudas and Visweswaran, 1990, 1993) to guarantee ϵ-global convergence are satisfied. Examining the new formulation reveals that if the y variables are held constant, then a linear objective function results subject to a linear set of constraints. If the $\boldsymbol{\mathcal{X}}$ variables are held fixed, then a convex objective function is obtained subject to a linear set of equality constraints.

The Primal Problem

The primal problem (S-P) is obtained by fixing the y variables such that $\boldsymbol{y} = \boldsymbol{y}^K \in Y$. To ensure feasibility, the mole fraction constraint ($\sum_i y_i^K = 1$) must be satisfied. (S-P) is explicitly given as:

$$\begin{aligned} \min_{\boldsymbol{\mathcal{X}}} \quad & \mathcal{C}(\boldsymbol{y}^K) + \sum_{i \in C} y_i^K \cdot \sum_{j \in C} \mathcal{G}_{ij} \tau_{ij} \mathcal{X}_j \\ s.t. \quad & \mathcal{X}_i \cdot \sum_{j \in C} \mathcal{G}_{ji} y_j^K = y_i^K \qquad \forall \quad i \in C. \end{aligned}$$

Note that (S-P) has a linear objective function and constraints in the variables $\mathcal{X}_i$. Its solution provides an upper bound on the global solution because it represents a restriction of the original problem. The definitions of the $\boldsymbol{\mathcal{X}}$ variables form a square set of equality constraints so that (S-P) is simply a function evaluation. This allows the Lagrange multipliers and the $\boldsymbol{\mathcal{X}}$ variable levels to be easily evaluated, as will be shown in the next section.

6.5.2 *Derivation of the Relaxed Dual Problem*

The primal problem yields an upper bound on the global solution. The relaxed dual subproblems will yield valid lower bounds.

6.5.2.1 Formulation of the Lagrange function. Crucial to the analysis is the construction of the Lagrange function for the problem, given as:

$$\begin{aligned} L(\boldsymbol{\mathcal{X}}, \boldsymbol{y}, \boldsymbol{\lambda}) &= \mathcal{C}(\boldsymbol{y}) + \sum_{i \in C} y_i \cdot \sum_{j \in C} \mathcal{G}_{ij} \tau_{ij} \mathcal{X}_j + \sum_{i \in C} \lambda_{\mathcal{X}_i} \left\{ \mathcal{X}_i \cdot \sum_{j \in C} \mathcal{G}_{ji} y_j - y_i \right\} \\ &= \sum_{i \in C} \mathcal{X}_i \left\{ \sum_{j \in C} \mathcal{G}_{ji} \left[\tau_{ji} + \lambda_{\mathcal{X}_i} \right] y_j \right\} + \mathcal{C}(\boldsymbol{y}) - \sum_{i \in C} y_i \lambda_{\mathcal{X}_i} \qquad (6.44) \end{aligned}$$

where $\lambda_{\mathcal{X}_i}$ are the multipliers associated with the bilinear constraints involving $\mathcal{X}_i$ of (S-N). The final form of Eqn. (6.44) has been obtained by simply collecting the $\boldsymbol{\mathcal{X}}$ variable terms.

Writing the Lagrangian for the primal as obtained from Eqn. (6.44) with $\boldsymbol{y} = \boldsymbol{y}^K$ yields:

$$L(\boldsymbol{\mathcal{X}}, \boldsymbol{y}^K, \boldsymbol{\lambda}) = \sum_{i \in C} \mathcal{X}_i \left\{ \sum_{j \in C} \mathcal{G}_{ji} \left[\tau_{ji} + \lambda_{\mathcal{X}_i} \right] y_j^K \right\} + \mathcal{C}(\boldsymbol{y}^K) - \sum_{i \in C} y_i^K \lambda_{\mathcal{X}_i} \quad (6.45)$$

and then invoking the first order gradient conditions gives:

$$\nabla_{\mathcal{X}_i} L(\boldsymbol{\mathcal{X}}, \boldsymbol{y}^K, \boldsymbol{\lambda}) = \sum_{j \in C} \mathcal{G}_{ji} \left[\tau_{ji} + \lambda_{\mathcal{X}_i} \right] y_j^K = 0 \qquad \forall \quad i \in C \quad (6.46)$$

$$\nabla_{\lambda_{\mathcal{X}_i}} L(\boldsymbol{\mathcal{X}}, \boldsymbol{y}^K, \boldsymbol{\lambda}) = \mathcal{X}_i \cdot \sum_{j \in C} \mathcal{G}_{ji} y_j^K - y_i^K = 0 \qquad \forall \ i \in C. \quad (6.47)$$

Eqns. (6.46) and (6.47) then provide the multipliers and the $\boldsymbol{\mathcal{X}}$ variables for any $\boldsymbol{y}^K$ as:

$$\lambda_{\mathcal{X}_i}^K = -\frac{\sum_j \mathcal{G}_{ji} \tau_{ji} y_j^K}{\sum_j \mathcal{G}_{ji} y_j^K} \qquad \forall \quad i \in C \quad (6.48)$$

$$\mathcal{X}_i^K = \frac{y_i^K}{\sum_j \mathcal{G}_{ji} y_j^K} \qquad \forall \quad i \in C. \quad (6.49)$$

Remark 6.5.2 Note that the complete solution to any primal problem is a simple set of function evaluations. Because the denominator of Eqn. (6.48) is always strictly greater than zero for any feasible $\boldsymbol{y}^K \in Y$, the Lagrange multipliers obtained from any primal problem are nonempty and bounded. This is a required condition of the GOP algorithm (Floudas and Visweswaran, 1990, 1993) to guarantee ϵ-global convergence.

6.5.2.2 Relaxation of the Lagrange function. The Lagrange function given by Eqn. (6.44) is now manipulated to generate valid lower bounds on the problem. The first step is to write the Lagrange function constructed using the multipliers of any given primal as:

$$L(\boldsymbol{\mathcal{X}}, \boldsymbol{y}, \lambda^K) = \sum_{i \in C} \mathcal{X}_i \left\{ \sum_{j \in C} \mathcal{G}_{ji} \left[\tau_{ji} + \lambda_{\mathcal{X}_i}^K \right] y_j \right\} + \mathcal{C}(\boldsymbol{y}) - \sum_{i \in C} y_i \lambda_{\mathcal{X}_i}^K. \quad (6.50)$$

The next step involves utilizing the KKT gradient conditions of the primal problem. Noting that λ^K obviously satisfies Eqns. (6.46), this set of equations is subtracted from the expression within curly braces of Eqn. (6.50) to obtain:

$$L(\boldsymbol{\mathcal{X}}, \boldsymbol{y}, \lambda^K) = \sum_{i \in C} \mathcal{X}_i \left\{ \sum_{j \in C} \mathcal{G}_{ji} \left[\tau_{ji} + \lambda_{\mathcal{X}_i}^K \right] \left[y_j - y_j^K \right] \right\} + \mathcal{C}(\boldsymbol{y}) - \sum_{i \in C} y_i \lambda_{\mathcal{X}_i}^K. \quad (6.51)$$

The relaxed dual problem also contains the normalizing mole fraction constraints. This constraint can be used to eliminate one of the y variables appearing in Eqn. (6.51). The last member of the set of components C is labeled $\bar{\imath}$, and this mole fraction is then defined as:

$$y_{\bar{\imath}} = 1 - \sum_{i \neq \bar{\imath}} y_i \quad (\text{and } y_{\bar{\imath}}^K = 1 - \sum_{i \neq \bar{\imath}} y_i^K) \tag{6.52}$$

with a new set, denoted CR, defined as the set of components excluding its last member:

$$CR \equiv C \setminus \{\bar{\imath}\}. \tag{6.53}$$

By substituting Eqn. (6.52) into the $\boldsymbol{y}$ variable expressions appearing within the curly braces of Eqn. (6.51), the Lagrangian can be written as follows:

$$\begin{aligned} L(\boldsymbol{\mathcal{X}}, \boldsymbol{y}, \lambda^K) &= \mathcal{C}(\boldsymbol{y}) - \sum_{i \in C} y_i \lambda_{\mathcal{X}_i}^K \\ &+ \sum_{i \in C} \mathcal{X}_i \left\{ \sum_{j \neq \bar{\imath}} \left\{ \mathcal{G}_{ji} \left[\tau_{ji} + \lambda_{\mathcal{X}_i}^K \right] - \mathcal{G}_{\bar{\imath}i} \left[\tau_{\bar{\imath}i} + \lambda_{\mathcal{X}_i}^K \right] \right\} \left\{ y_j - y_j^K \right\} \right\}. \end{aligned} \tag{6.54}$$

The derivatives of Eqn. (6.54) with respect to the $\boldsymbol{\mathcal{X}}$ variables are the qualifying constraints of the GOP and are given as follows:

$$\begin{aligned} g_i(\boldsymbol{y}) &= \nabla_{\mathcal{X}_i} L(\boldsymbol{\mathcal{X}}, \boldsymbol{y}, \lambda^K) \\ &= \sum_{j \neq \bar{\imath}} \left\{ \mathcal{G}_{ji} \left[\tau_{ji} + \lambda_{\mathcal{X}_i}^K \right] - \mathcal{G}_{\bar{\imath}i} \left[\tau_{\bar{\imath}i} + \lambda_{\mathcal{X}_i}^K \right] \right\} \left\{ y_j - y_j^K \right\} \quad \forall \quad i \in C. \end{aligned} \tag{6.55}$$

Remark 6.5.3 Note that these are hyperplanes in the y variable space that can be employed to partition the solution domain into regions. If these constraints are allowed to be of constant sign (i.e. $g_i(\boldsymbol{y}) \geq 0$ or $g_i(\boldsymbol{y}) \leq 0$ for each i), then based on the sign of each qualifying constraint, it is possible to set bounds on the $\boldsymbol{\mathcal{X}}$ variables so that the Lagrangian will validly underestimate the solution in the partition defined by the set of qualifying constraints. For example, if $g_i(\boldsymbol{y}) \geq 0$, then the corresponding $\mathcal{X}_i$ may be set to its lower bound, $\mathcal{X}_i^L$. This will ensure that $\mathcal{X}_i^L \cdot g_i(\boldsymbol{y}) \leq \mathcal{X}_i \cdot g_i(\boldsymbol{y})$ in that particular partition of the $\boldsymbol{y}$ variable space.

6.5.2.3 Augmentation of the $\mathcal{X}$ variable set. Rather than having qualifying constraints that partition the $\boldsymbol{y}$ variable space by hyperplanes, it is preferable to have a simpler form of qualifying constraint. This is achieved by augmenting the set of $\boldsymbol{\mathcal{X}}$ variables, with the new set denoted as $\hat{\boldsymbol{\mathcal{X}}}$, defined below:

$$\text{For each } i \in C, \quad \text{define} \quad \hat{\mathcal{X}}_{ij} = \mathcal{X}_i \quad \forall \; j \in CR. \tag{6.56}$$

This allows the Lagrangian to be written as follows:

$$\begin{aligned} L(\hat{\boldsymbol{\mathcal{X}}}, \boldsymbol{y}, \lambda^K) &= \mathcal{C}(\boldsymbol{y}) - \sum_{i \in C} y_i \lambda^K_{\mathcal{X}_i} \\ &+ \sum_{i \in C} \sum_{j \neq \bar{\imath}} \hat{\mathcal{X}}_{ij} \left\{ \left\{ \mathcal{G}_{ji} \left[\tau_{ji} + \lambda^K_{\mathcal{X}_i} \right] - \mathcal{G}_{\bar{\imath}i} \left[\tau_{\bar{\imath}i} + \lambda^K_{\mathcal{X}_i} \right] \right\} \cdot \left\{ y_j - y^K_j \right\} \right\}. \end{aligned} \tag{6.57}$$

The effect of the above step is seen in the new qualifying constraints now defined as follows:

$$\begin{aligned} \hat{g}_{ij}(\boldsymbol{y}) &= \nabla_{\hat{\mathcal{X}}_{ij}} L(\hat{\boldsymbol{\mathcal{X}}}, \boldsymbol{y}, \lambda^K) \\ &= \left\{ \mathcal{G}_{ji} \left[\tau_{ji} + \lambda^K_{\mathcal{X}_i} \right] - \mathcal{G}_{\bar{\imath}i} \left[\tau_{\bar{\imath}i} + \lambda^K_{\mathcal{X}_i} \right] \right\} \cdot \left\{ y_j - y^K_j \right\} \ \forall\, i \in C, j \in CR. \end{aligned} \tag{6.58}$$

These qualifying constraints all share the same *single* term $(y_j - y^K_j)$, allowing the $\boldsymbol{y}$ variable space to be partitioned into simple orthogonal regions. This is described in the next section.

The partitioning of the $\boldsymbol{y}$ variable space, establishing and setting the bounds on the $\mathcal{X}$ variables is done in a similar way as in the minimization of the Gibbs free energy problem that we discussed in section 4.2, and it is described in detail in McDonald and Floudas (1995a). The global optimization algorithm for the NRTL model has also a similar structure to the one for the minimization of the Gibbs free energy and can be found in McDonald and Floudas (1995a).

6.6 ILLUSTRATIVE EXAMPLE FOR THE NRTL : N-BUTYL-ACETATE – WATER

Prior to presenting the illustrative example the following points should be mentioned:

(i) The postulated solutions used to test for stability are obtained from minimizing the Gibbs free energy function, either locally or from the global optimization approach presented earlier. These solutions have a separate notation associated with them. The set of phases is denoted $P \equiv P_L \cup P_V \equiv \{k\}$, where the liquid phases are designated $L_k \subseteq P_L$, and the vapor phase is denoted as $V \equiv P_V$. For any postulated solution, the mole fractions of each component i in some phase k are given by z^k_i, so that $\sum_i z^k_i = 1 \ \forall\, k \in P$. The parameter ϕ^k corresponds to the fraction of total mass in phase k, so that $\sum_k \phi^k = 1$.

(ii) In the results, y^*_i is the mole vector at a global solution, with $\mathcal{F}^*$ being the corresponding tangent plane distance function. The numbers in parentheses below $\mathcal{F}^*$ show the results of using the NLP solver MINOS 5.4 to solve the original nonconvex formulations from 100 randomly chosen starting points, with the first number corresponding to the number of failures in obtaining a negative tangent plane distance function when the postulated solution is a non-equilibrium one, and the second number showing how many times the global solution was obtained. N_I is the number of iterations, with the

number in parentheses corresponding to the iteration number at which the tangent plane distance function becomes negative. The number in parentheses reported under the cpu time for the problems corresponds to the total cpu time consumed before a negative tangent plane distance is obtained. N_F is the percentage of solution nodes that are fathomed because the lower bound lies above the best current upper bound.

(iii) All computational runs were performed on a Hewlett Packard 9000/730 machine. The algorithm is implemented in C as the package GLOPEQ (GLObal OPtimization for the Phase EQuilibrium problem), (McDonald and Floudas, 1997). The solver MINOS 5.4 is accessed as a subroutine. All CPU times reported represent the total *real time* taken to solve any given phase stability problem to ϵ-global optimality, with a tolerance of $\epsilon_a = 10^{-6}$.

This example is small but challenging and is analyzed here at four different feed conditions. There are two components so that the phase rule allows a maximum of two phases and these are both assumed to be liquid phases. This implies that the Gibbs free energy of vaporization terms may be left out of the expression for the tangent plane distance function as indicated in Comment (i) above. The binary parameters of the system are given by Heidemann and Mandhane (1973) as:

$$\tau_{12} = 3.00498\ ,\ \tau_{21} = 4.69071\ ,\ \alpha_{12} = 0.391965.$$

The explicit formulation for the phase stability problem expressed in terms of the original and transformed variables is as follows:

$$\begin{aligned}
\min \quad & y_1 \ln y_1 + y_2 \ln y_2 - y_1\mu_1^0(z) - y_2\mu_2^0(z) + \mathcal{G}_{12}\tau_{12}y_1\mathcal{X}_2 + \mathcal{G}_{21}\tau_{21}y_2\mathcal{X}_1 \\
s.t. \quad & \mathcal{X}_1 \cdot \{y_1 + \mathcal{G}_{21}y_2\} = y_1 \\
& \mathcal{X}_2 \cdot \{\mathcal{G}_{12}y_1 + y_2\} = y_2 \\
& y_1 + y_2 = 1 \\
& 0 \le y_1, y_2 \le 1.
\end{aligned}$$

Note that there are two bilinear terms in the objective function, and two bilinear constraints, which make the problem nonconvex. The Lagrange takes the form:

$$\begin{aligned}
L(\boldsymbol{\mathcal{X}}, \boldsymbol{y}, \lambda^K) \;=\; & y_1 \ln y_1 + y_2 \ln y_2 - y_1\left[\mu_1^0(z) + \lambda_1^K\right] - y_2\left[\mu_2^0(z) + \lambda_2^K\right] \\
& + \mathcal{X}_1\left\{\lambda_1^K - \mathcal{G}_{21}\left[\tau_{21} + \lambda_1^K\right]\right\}\left\{y_1 - y_1^K\right\} \\
& + \mathcal{X}_2\left\{\mathcal{G}_{12}\left[\tau_{12} + \lambda_2^K\right] - \lambda_2^K\right\}\left\{y_1 - y_1^K\right\}.
\end{aligned}$$

The first line of the Lagrangian contains convex and linear terms, while the second contains bilinear terms in $\boldsymbol{\mathcal{X}}$ and $\boldsymbol{y}$.

Geometric Interpretation The performance of the GOP will now be illustrated for the case of an equimolar feed, so that the total number of mols of each

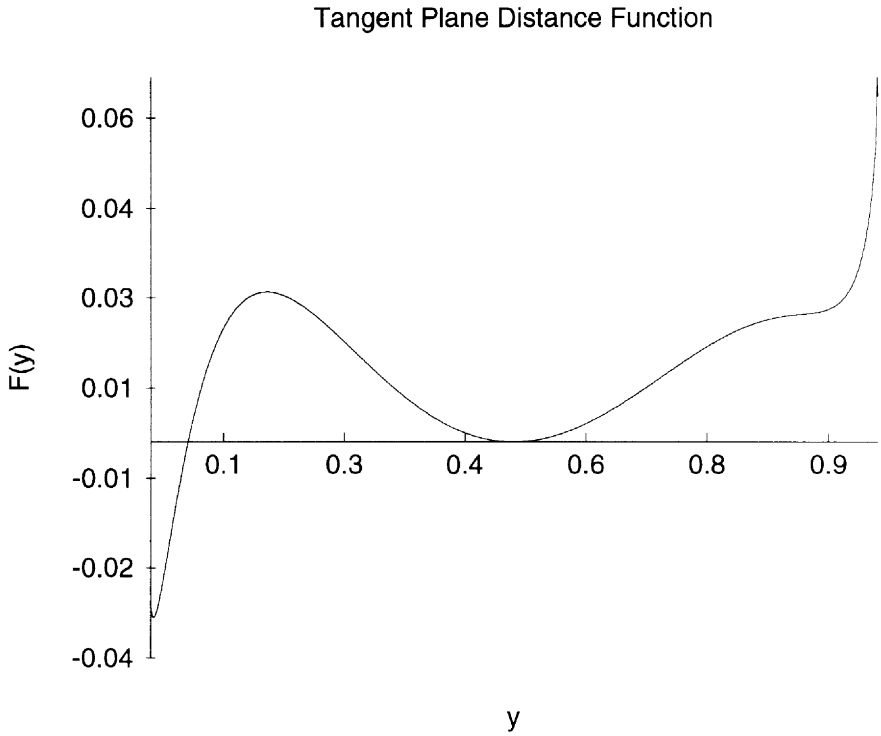


Figure 6.11. Plot of $\mathcal{F}(y_1)$ for Example 1

component is given as $n_1^T = n_2^T = 0.5$. The test candidate phase will be a single liquid phase so that $z_1 = z_2 = 0.5$. These values of z yield $\mu_1^0(z) = -0.06391$ and $\mu_2^0(z) = 0.02875$. The objective function is plotted in Figure 6.11 and reveals a nonconvex curve with multiple stationary and inflection points. In fact, there are two local minima, one of which is the global one, a local maximum and several inflection points, and the values of y_1 where these occur (excluding the maxima at the boundaries of the feasible region) are given in Table 6.10. A local solver will converge at best to a local minimum, with no guarantee of obtaining the global solution. For this case, if the global solution is not obtained, then all the other stationary points yield a nonnegative objective function value, which would lead to the erroneous conclusion that the single liquid phase is stable. In fact, when solving this problem using MINOS 5.4, only 29 of 100 randomly selected starting points gave the global solution. This clearly demonstrates the difficulties associated with the phase and chemical equilibrium problem and highlights the need for a global optimization approach. The manner in which the GOP algorithm proceeds is now demonstrated for the first two iterations.

Nature of SP/IP	y_1
Global Minimum	0.0042
Local Minimum	0.500
Local Maximum	0.160
Inflection Point	0.025
Inflection Point	0.279
Inflection Point	0.706
Inflection Point	0.895

Table 6.10. Stationary points for Example 1

Initialization: Choose an initial point of $y_1^1 = y_2^1 = 0.5$.

Iteration 1: The initial point gives the upper bound as $\mathcal{F}(\boldsymbol{y}^1) = 0$. The multipliers are calculated using Eqns. (6.48) and the premultiplying terms of the qualifying constraints are also calculated in order to obtain $\hat{s}_i^K$:

$$\begin{aligned} \lambda_1^1 = -0.644 \quad &\Rightarrow \quad \lambda_1^1 - \mathcal{G}_{21}[\tau_{21} + \lambda_1^1] = -1.287 \Rightarrow \hat{s}_1^1 = -1 \\ \lambda_2^1 = -0.707 \quad &\Rightarrow \quad \mathcal{G}_{12}[\tau_{12} + \lambda_2^1] - \lambda_2^1 = \quad 1.415 \Rightarrow \hat{s}_2^1 = +1. \end{aligned}$$

The parent region is divided in two, and a relaxed dual subproblem is solved in each.

Region 1: The box bounds are: $0.5 \leq y_1 \leq 1.0$ (and hence $0.0 \leq y_2 \leq 0.5$). This implies that $(y_1 - y_1^1) \geq 0$. The combined sign of the premultiplying term of $\mathcal{X}_1$ is therefore negative so that it is set to its upper bound of 1, while $\mathcal{X}_2$ is set to its lower bound of 0. Solving the relaxed dual problem in this region yields a global solution of $y_1^* = 0.7788$ with a corresponding lower bound of $\mu_1^* = -0.1861$.

Region 2: The box bounds are: $0.0 \leq y_1 \leq 0.5$ (and hence $0.5 \leq y_2 \leq 1.0$). This implies that $(y_1 - y_1^1) \leq 0$, so that $\mathcal{X}_1$ is set to its lower bound of 0 and $\mathcal{X}_2$ takes on its upper bound of 1. The global solution of the relaxed dual is $y_1^* = 0.1910$ with a corresponding lower bound of $\mu_2^* = -0.2469$.

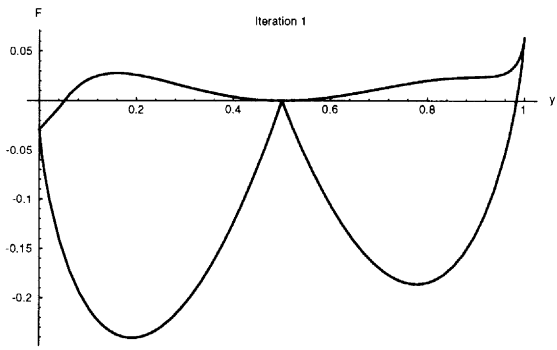


Figure 6.12. Plot of $\mathcal{F}$ and underestimating Lagrangians for Iteration 1

The underestimators for Regions 1 and 2 are drawn in Figure 6.12. Notice their convex shape and the fact that they touch the tangent plane distance function at the current point. Both lower bounds from the first iteration are lower than the best current upper bound and are added to the set of relaxed dual solution nodes.

<u>**Iteration 2**</u>: The infimum of all the available lower bounds supplies Region 2 of Iteration 1 as the parent region, with $y_1^2 = 0.1910$. The primal problem gives $\mathcal{F}(\boldsymbol{y}^2)$ as 0.0267 so that the best upper bound remains zero. The multipliers are then:

$$\begin{aligned} \lambda_1^2 = -1.888 \quad &\Rightarrow \quad \lambda_1^2 - \mathcal{G}_{21}[\tau_{21} + \lambda_1^2] = -2.334 \Rightarrow \hat{s}_1^2 = -1 \\ \lambda_2^2 = -0.204 \quad &\Rightarrow \quad \mathcal{G}_{12}[\tau_{12} + \lambda_2^2] - \lambda_2^1 = \quad 1.066 \Rightarrow \hat{s}_2^2 = +1. \end{aligned}$$

Again two relaxed dual subproblems are solved with solution levels as follows:

$$\begin{array}{lllllll} \text{Region 1:} & 0.1910 & \leq & y_1 & \leq & 0.5 & \Rightarrow \quad y_1^* = 0.3592 \quad \mu_3^* = -0.0413. \\ \text{Region 2:} & 0.0 & \leq & y_1 & \leq & 0.1910 & \Rightarrow \quad y_1^* = 0.0550 \quad \mu_4^* = -0.0853. \end{array}$$

These lower bounds are added to the set of relaxed dual solution nodes. For these subproblems two Lagrange functions are required. The $\boldsymbol{\mathcal{X}}$ variable bounds

Solutions for n-Butyl-Acetate (1) – Water (2) at $T, P = 1$ atm							
Component	Feed (moles)	$\mu_i^0(z)$ (—)	y^* (—)	$\mathcal{F}^*$ (—)	cpu (sec)	N_I (—)	N_F (%)
$C_6H_{12}O_2$ (1)	0.50	−0.06391	0.00421	−0.03247	0.06	12	33
H_2O (2)	0.50	0.02875	0.99579	(71/29)	(0.02)	(4)	
$C_6H_{12}O_2$ (1)	0.10	0.18825	0.96345	−0.21419	0.07	14	50
H_2O (2)	0.90	0.02412	0.03655	(0/5)	(0.001)	(1)	
$C_6H_{12}O_2$ (1)	0.20	−0.10252	0.00380	−0.07427	0.05	9	28
H_2O (2)	0.80	0.07088	0.99620	(0/100)	(0.01)	(1)	
$C_6H_{12}O_2$ (1)	0.65	−0.02708	0.94131	−0.00671	0.11	22	48
H_2O (2)	0.35	−0.01910	0.05869	(87/13)	(0.001)	(4)	

Table 6.11. Data and solutions for Example 1 with one liquid phase postulated

are set in different ways for each of them and the situation for Region 1 is now detailed. $(y_1 - y_1^2) \geq 0$ so that $\mathcal{X}_1^2 = \mathcal{X}_1^U = 0.863$ and $\mathcal{X}_2^2 = \mathcal{X}_2^L = 0.765$, where the superscript corresponds to the iteration number. The region associated with the parent of the current node provides the Lagrangian from Region 2 of Iteration 1, where $(y_1 - y_1^1) \leq 0$, implying that $\mathcal{X}_1^1 = \mathcal{X}_1^L = 0.597$ and $\mathcal{X}_2^1 = \mathcal{X}_2^U = 0.932$ for this Lagrangian.

At the third iteration, Region 1 of Iteration 1 yields the next parent region, so that again 2 relaxed dual subproblems are solved to give two additional lower bounds. The algorithm proceeds in this way until the infimum of all remaining lower bounds is within ϵ_a of the current best upper bound. For this particular problem, convergence occurs after 12 iterations. The total time taken by the algorithm was 0.06 CPU sec., and 8 nodes were fathomed. The solution is given in Table 6.11. Note also that $\mathcal{F}^*$ becomes negative at iteration 4 after 0.02 CPU secs. At this point, the algorithm could be terminated as it is clear that the postulated equilibrium solution is unstable.

Other feed conditions The algorithm was also tested for three other feed conditions, with different values of $\{n_i^T\}$. The stability test was initiated in

Solutions for n-Butyl-Acetate (1) – Water (2) for all feed conditions									
Comp.	Solution	$z_i^{L_1}$ (—)	$z_i^{L_2}$ (—)	$\mu_i^0(z)$ (—)	y^* (—)	$\mathcal{F}^*$ (—)	cpu (sec)	N_I (—)	N_F (%)
$i=1$	Local LL	0.93514	0.00456	−0.03523	0.59425	−0.00070	0.15	32	45
$i=2$	$\phi^{L_1}=0.21001$	0.06486	0.99544	−0.00398	0.40575	(25/75)	(0.002)	(1)	
$i=1$	Global LL	0.59199	0.00456	−0.03642	0.59199	0.0	0.16	36	43
$i=2$	$\phi^{L_1}=0.33271$	0.40801	0.99544	−0.00398	0.40801	(-)	(-)	(-)	

Table 6.12. Data and global solutions for Example 1

all cases by a single liquid phase solution, with $z_i = n_i^T / \sum_j n_j^T \ \forall\, i$. The only difference between the formulations will then be the values calculated for $\mu_i^0(z)$. This leads to negative tangent plane distances in all cases and these results are also shown in Table 6.11. The first and last feed conditions ($n_1^T = 0.5$ and $n_1^T = 0.65$) are seen to be the most difficult. The local searches show a large number of failures in terms of predicting that the one phase solution is unstable. This is also reflected in the fact that a negative tangent plane distance is not obtained until the fourth iteration in both cases. On the other hand, the middle two feed conditions are much less challenging, with the local searches showing no failures in predicting instability. However, for $n_1^T = 0.1$, the global solution was obtained only 5 times out of 100.

Local LL solution When the Gibbs free energy is minimized to attempt to obtain the global solution, the four feed conditions yield a strong local LL solution which satisfies the first order conditions of equal chemical potentials. This solution is given in Table 6.12 and can be used to test the algorithm. The mole fractions in the two phases are the same for all four feed conditions as they lie along the same tieline so that only the phase fractions differ. Obviously the tangent plane distance function is negative. As for the illustrative example, notice that if a local solver converges to a stationary point other than the global minimum, then it will incorrectly confirm a false equilibrium solution. The global minimum is at $y_1^* = 0.59425$ with $\mathcal{F}(\boldsymbol{y}^*) = -0.0007$. The algorithm takes 30 iterations to converge to this solution using a total of 0.15 CPU sec., while 45% of the nodes were fathomed.

Global LL solution All four feed conditions share the same mole fractions for the global solution, and these are also supplied in Table 6.12. The postulated global solution will obviously yield a nonnegative tangent plane distance over

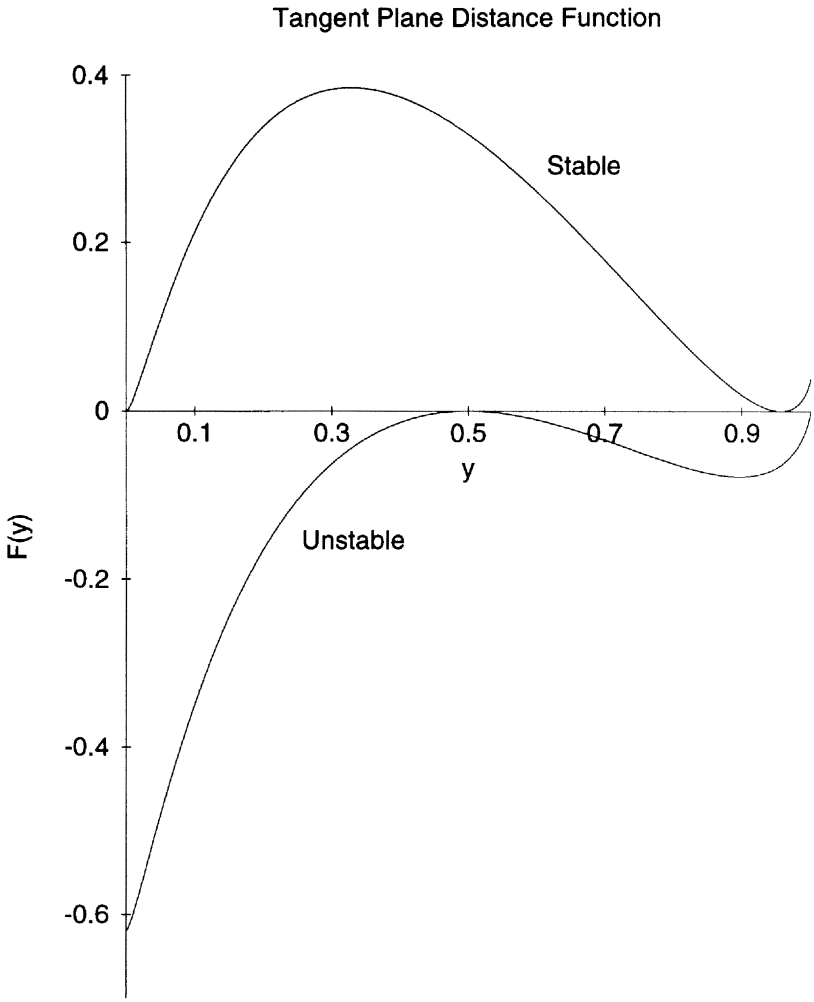


Figure 6.13. Plot of $\mathcal{F}$ for Example 1, trivial and global solutions postulated

the full mole fraction space and this is pictured in Figure 6.13. For a starting point of $y_1^1 = 0.75$, the global solution of zero is obtained after 36 iterations with the algorithm consuming a total of 0.16 CPU sec., while 43% of the nodes were fathomed. The time taken to verify the equilibrium solution is not significantly greater than that taken to establish instability for the other non-equilibrium postulated solutions. It should be noted that the iteration at which the tangent plane distance function becomes negative occurs in the first iterations. This will have important implications when the tangent plane criterion is used to test solutions obtained from the minimization of the Gibbs free energy because the concern is determining if this distance is negative.

Summary

In this chapter, we discussed the application of the GOP principles for the global optimization of important problems in phase and chemical reaction equilibrium such as (i) the minimization of the Gibbs free energy and (ii) the tangent plane stability analysis criterion. The expository scope of the subject focused on the NRTL activity coefficient model. For analysis and global optimization of problems (i) and (ii) employing other activity coefficient models such as T-K Wilson, ASOG, UNIQUAC, and UNIFAC the reader is directed to McDonald and Floudas (1994), (1995a), (1995b), (1995c), and (1997). Recently, we have developed rigorous global optimization methods for both the minimization of the Gibbs free energy and the tangent plane stability problems for a variety of equation of state models (Harding and Floudas (1999)). These methods are based on novel mathematical representations and on the αBB global optimization principles that are discussed in Part III of this book.

7 THE GOP APPROACH : DISTRIBUTED IMPLEMENTATION

In this chapter, we discuss the parallel implementation of the GOP algorithm, as studied by Androulakis *et al.* (1996), and we present distributed computing results for indefinite quadratic programming problems and for large-scale pooling problems that arise in chemical refineries. Section 7.1 focuses on the critical components of the distributed implementation of the GOP approach. Section 7.2 presents the computational results for large-scale indefinite quadratic problems. Finally, section 7.3 discusses the computational performance of the parallel GOP approach for large-scale blending and pooling problems.

7.1 CRITICAL COMPONENTS OF THE DISTRIBUTED GOP APPROACH

Androulakis *et al.* (1996) studied the distributed computing issues for the GOP algorithm and demonstrated that large-scale quadratic programming and large scale pooling problems can be addressed. They identified the following three major sources of computational challenges : (i) the update of the bounds on the connected $\boldsymbol{x}$ variables, (ii) the solution of the relaxed dual problems, and (iii) the routing of the appropriate data.

The computational difficulty of the GOP algorithm manifests itself in the solution of 2^{NI_c} problems, where NI_c is the number of connected variables. The connected $\boldsymbol{x}$ variables form a subset of the original $\boldsymbol{x}$–type variables. It was shown theoretically (Visweswaran and Floudas (1993)), and observed compu-

C. A. Floudas, *Deterministic Global Optimization*
© Springer Science+Business Media Dordrecht 2000

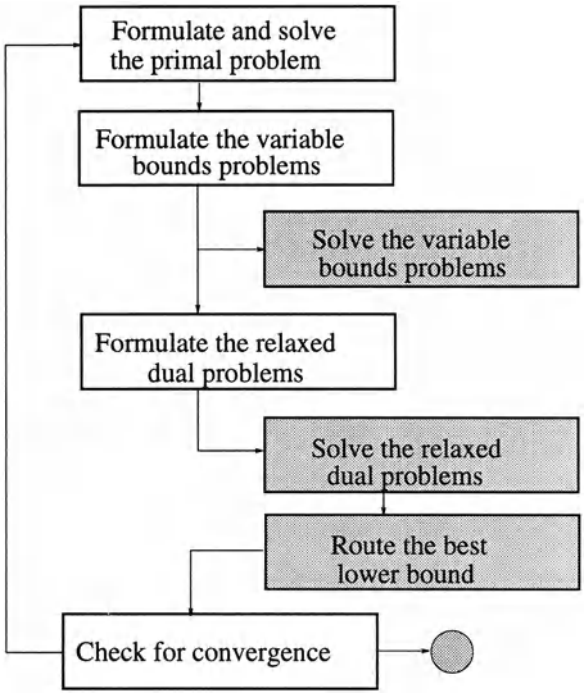


Figure 7.1. Flow Diagram of distributed GOP

tationally that obtaining tight bounds on the optimization variables, for both the $\boldsymbol{x}$–type as well as the $\boldsymbol{y}$–type, is very helpful in the convergence rate of the algorithm. In order to calculate tight variable bounds one has to solve $2(N_x + N_y)$ convex NLP's, where N_x and N_y if the total number of $\boldsymbol{x}$–type and $\boldsymbol{y}$–type variables, respectively. Therefore, the search for tighter variable bounds problems can be computationally improved if these problems are solved in parallel.

The major computational bottleneck of the method is the solution of a potentially very large number of relaxed dual problems at a given iteration, 2^{NI_c}. Therefore, major emphasis has to be placed on the most efficient solution of this large number of convex or linear optimization problems.

Finally, issues related to the routing of the appropriate data, once a lower bound has been identified, will also be addressed. Such issues require the implementation of a parallel *routing/sorting* algorithm. Figure 7.1 depicts the basic steps of the distributed implementation of the GOP algorithm and highlights the parallelized steps.

7.1.1 Updating the Variable Bounds

In order to identify the tightest possible variable bounds, we have to calculate the maximum and minimum possible values of all the variables within the current domain of interest. Based on the partitioning induced by the GOP algorithm, the domain of interest for the solution of the relaxed dual problem, is defined by : (a) the original convex constraints, (b) the original convexified constraints, and (c) the previous qualifying constraints. Constraints (a) and (b), define the range of variables with respect to the original problem. Any convex constraint, that is a convex inequality and/or an affine equality, will not alter the convexity characteristics of the problem and thus can be used. Any convexification of the original nonconvex constraints will be an overestimation of the feasible region, and it would restrict the domain for the purpose of identifying tighter variable bounds. In addition, the current domain of interest, over which the new lower bound will be sought, is implicitly defined by the set of the previous qualifying constraints.

It was also observed computationally that the frequency at which these problems are solved can be treated as a decision variable. For certain classes of problems, (e.g., indefinite quadratic), computing tight bounds once at the very beginning was adequate, whereas for other classes of problems (e.g., pooling and blending), the variable bounds had to be updated at each iteration. It is evident that the total number of variable bounds problems that have to be solved are $2(N_x + N_y)$, implying that for large scale optimization problems the framework of distributed computing is needed. With respect to the implementation, it is first identified whether it is worth solving the bounds problems in parallel. Then, the vector of variables is divided into smaller groups and these groups are assigned to nodes that are responsible for solving the variable bounds problems associated with variables. The master node then collects the partial vector. The collection process has an unavoidable sequential character but the gains from solving the variable bounds in parallel outweigh any potential performance degradation.

7.1.2 Solving the Relaxed Dual Problems

The parallel solution of the relaxed dual problems aims at addressing the need to reduce the computational burden associated with the solution of 2^{NIc} problems at each iteration.

Based on the theoretical analysis of the method, it is clear that all the relaxed dual problems that need to be solved, have the same functional form, and only the bound combinations of the x–type variables will be different. Therefore, what distinguishes one relaxed dual problem problem from the others is the bound combination at which the linearization is computed, and the qualifying constraints that have to be included. As can be seen in Figure 7.2, the y–domain is partitioned based on the signs of the qualifying constraints. In this simple illustration we assume that there exist two connected variables that give rise to four bound combinations, that is four possible sign combinations

of the qualifying constraints. A particular node in the parallel architecture is responsible for solving the primal problem and preparing all the necessary data for the formulation of the relaxed dual problems. Subsequently, each node determines whether it is responsible for solving any relaxed dual problems. The next step is, for every node, to proceed to solve the relaxed dual problems corresponding to the bound combinations that have been assigned to it. Once the assigned problems have been solved, all the feasible solutions are stored in the local CPU's memory and only the best lower bound generated at each processing element is propagated to the "master" node. This issue brings us naturally to the third issue associated with the distributed implementation of the GOP algorithm, that is the routing of the best lower bound.

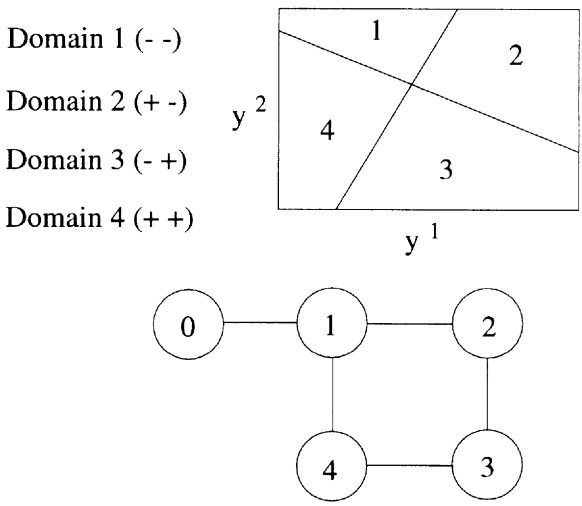


Figure 7.2. Parallel Solution of the Relaxed Dual Problems.

7.1.3 Routing of the Best Lower Bound

Poor data communication in parallel architectures can create substantial bottlenecks, thus degrading the overall performance of the algorithm. For the "master" node to proceed with the solution of the next primal problem only information related to the best lower bound is needed. Furthermore, it is rare to envision a situation in which hundreds of processing elements attempt to, almost simultaneously, access a particular node in order to provide certain data. The queuing problems that would arise will be very significant. Therefore, a routing algorithm was implemented which would, in $\lfloor log(P) + 1 \rfloor$ steps, where P is the number of nodes, transmit to node 0 the best lower bound. This is described in detail in Androulakis et al. (1996).

7.2 LARGE SCALE INDEFINITE QUADRATIC PROBLEMS

In this section, we will consider the application of the distributed version of the GOP algorithm, in an *Intel-Paragon* machine, to large scale indefinite quadratic optimization problems. The generic formulation of Pardalos and Rosen (1988) is considered. According to that formulation the optimization problem is stated as follows:

$$\begin{aligned}
&\min_{\boldsymbol{x},\boldsymbol{y}\in\Omega} \psi(\boldsymbol{x},\boldsymbol{y}) = \sum_{i=1}^{n} \lambda_i (x_i - \overline{\omega_i})^2 + \boldsymbol{d}^T \boldsymbol{y} \\
&\Omega = \{(\boldsymbol{x},\boldsymbol{y}) : \boldsymbol{A}_1\boldsymbol{x} + \boldsymbol{A}_2\boldsymbol{y} \leq \boldsymbol{b}\ , \quad \boldsymbol{x} \geq \boldsymbol{0}, \boldsymbol{y} \geq \boldsymbol{0}\} \\
&\boldsymbol{x}, \lambda, \overline{\omega} \in \Re^n \\
&\boldsymbol{y}, \boldsymbol{d} \in \Re^k \\
&\boldsymbol{A}_1 \in \Re^{m\times n}, \boldsymbol{A}_2 \in \Re^{m\times k}.
\end{aligned} \tag{7.1}$$

By construction, we generate half of the eigenvalues, λ_i, positive and half negative. This is the most difficult problem to address since the solution, unlike strictly concave problems, may not lie on a vertex point. Several runs, for different problem sizes, were performed and the results are analyzed with respect to (i) the effect of the linear constraints and linear variables; (ii) the effect of linear constraints; (iii) the effect of linear variables. Finally, some very computationally intensive tasks are discussed. In all runs we denote by k the number of linear variables, m the number of linear constraints, and n the number of quadratic variables.

7.2.1 *Combined Effect of the Linear Constraints and Linear Variables*

The first set of computational results aims at demonstrating the effect on the performance of the GOP for different values of k and m. Typical results are shown in Table 7.1.

Based on these results the following observation can be made : *as long as* $\frac{k+m}{n} \geq 1$ *and* $\frac{m}{n} \geq 1$ *the GOP algorithm identifies the global minimum with maximum efficiency.* The GOP algorithm takes two iterations and solves only two primal problems and two relaxed dual problems. Qualitatively, this implies that, for this particular structure of problems, such a combination of the parameters forces the solution to lie close to a vertex point. The GOP algorithm identifies that fact very efficiently and converges in a minimum number of iterations.

7.2.2 *Effect of the Number of Linear Constraints*

The next computational experiment aims at isolating the effect of the linear constraints in the behavior of GOP. The computational results have been tabulated in Table 7.2 for constant values of $k = 200$, and $n = 100$.

As it can be seen, increasing the number of linear constraints results in an increase in the computational effort initially. This can be seen by observing both NI_c and the number of the required iterations. As m increases, the

k	m	n	N	M	CPU(s)
80	120	100	180	480	4.49
100	100	100	200	500	5.42
100	200	200	300	800	10.3
100	300	100	200	700	9.38
150	150	250	400	950	12.3
200	200	100	300	800	6.26
200	200	200	400	1000	7.86
200	200	220	420	1040	17.4
200	200	250	450	1100	4.50
200	300	100	300	900	5.08
300	300	100	400	1100	9.58
400	200	100	500	1200	13.2

Table 7.1. Combined Effect of m and k. N = m+n, M = m + 2(k+n).

m	NI_c	Itn	PE	CPU(s)
1	0	2	2	2.45
5	5	18	32	30.1
10	9	30	32	96.3
20	8	4	32	44.5
40	8	4	32	60.1
60	3	4	4	11.3
100	0	2	2	5.42
200	0	2	2	5.76
300	0	2	2	7.62

Table 7.2. Effect of m. n = 100, k = 200.

addition of linear constraints in the feasible region increases the number of vertices in the polytope and as a result the GOP becomes very efficient in determining the global minimum. The results are also depicted in Figure 7.3.

7.2.3 Effect of the Number of Linear Variables

The effect of increasing the number of linear variables on the behavior of the GOP is more complicated. The computational results of Table 7.3 indicate the complex nature of the effect of the value of k.

It is clear from Figure 7.4 that k has a nonmonotonic effect and eventually, the larger the value of k the more difficult the problem becomes. It should be pointed out that the difficulty, in terms of GOP, is expressed through the

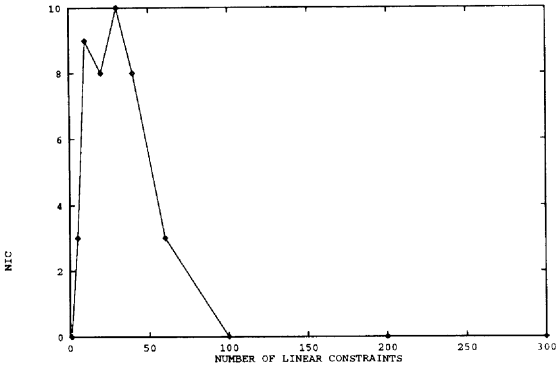


Figure 7.3. Effect of m

k	NI_c	Itn	PE	CPU(s)
0	1	2	2	3.17
20	1	2	2	3.80
40	3	2	2	6.30
60	3	2	4	11.3
80	4	2	8	28.1
90	1	2	4	7.45
100	1	2	2	5.42
130	5	2	32	32.2
150	6	2	32	41.8
200	6	2	32	54.2
300	9	3	32	192
400	10	2	32	198

Table 7.3. Effect of k. n = 100, m = 100.

number of connected variables, NI_c, and the number of iterations for the GOP to converge. The CPU is an indication as well, but for the large problems of Section 7.2.1 the large CPU times were due to the large size of the resulting convex sub-problems that had to be solved.

7.2.4 Effect of the Number of Nonlinear Variables

Finally, the last set of computational experiments deals with certain instances which are computationally very intensive in terms of the connected variables and number of iterations. The results have been summarized in Table 7.4.

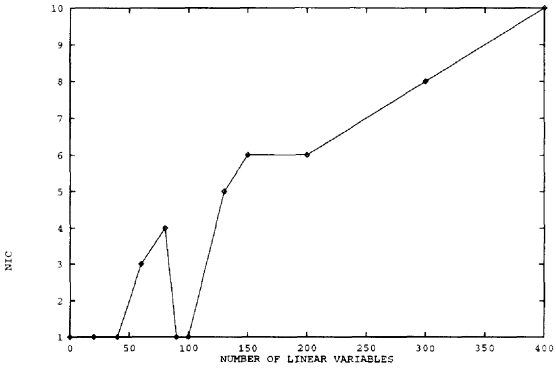


Figure 7.4. Effect of k

k	m	n	NI_c	Itn	PE	CPU(s)
30	20	100	9	54	32	100.
20	20	150	7	7	64	14.0
20	20	200	16	60	64	1847
300	100	200	8	3	32	54.0
50	50	220	10	3	64	31.6
50	50	250	14	16	64	796.
50	50	275	16	3	64	1800
50	50	300	17	3	64	3260
75	75	300	15	3	64	942.
75	75	350	11	3	64	209.

Table 7.4. Some computationally intensive tasks.

It is important to note from Table 7.4 that although the absolute size of the problems might not be that large ($k = 30$, $m = 20$, and $n = 100$ for instance) the difficulty of the problem can be observed both in terms of the number of iterations required and in terms of the number of connected variables. Note that for all the representative runs of Table 7.4 the relations between k, m, and n that define an "*easy*" problem for GOP are violated. By combining the theoretical advances of the GOP along with the distributed implementation of the algorithm problems of large size can be addressed.

Finally, we observe the computational requirements, in terms of the total CPU time, as a function of the the number of connected variables. As can be seen from Figure 7.5 there exists a linear relationship between the logarithm of

the CPU time and the number of connected variables, NI_c as expected since the number of relaxed duals increases exponentially with NI_c.

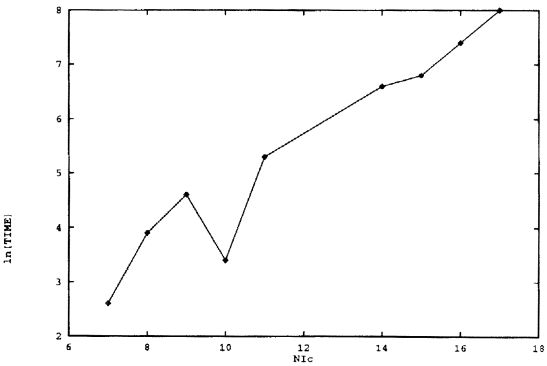


Figure 7.5. Time vs. NI_s

From the computational results for indefinite quadratic problems we observe that :

- problems of 400 linear variables, 100 nonlinear variables, 200 linear constraints, and 500 bound constraints can be solved in 13.2 sec. as shown in Table 7.1;
- increasing the size of the linear constraints makes the problem easier for the GOP (e.g., problems with 300 linear constraints, 200 linear variables, 100 nonlinear variables require 7.6 sec.);
- increasing the number of linear variables to 400 while maintaining 100 nonlinear variables and 100 constraints increases the CPU to 198 sec.; and
- problems that correspond to increasing the number of nonlinear variables up to 350 can still be solved with reasonable computational effort, as it is shown in Table 7.4.

7.3 LARGE SCALE POOLING AND BLENDING PROBLEMS

In this section we revisit and discuss the solution of a specific formulation of pooling and blending problems, that we studied in section 4.2.3 of Chapter 4, using the GOP. Such problems are very often encountered in various chemical processes. They describe the situation in which a set of components (*ncomp*) with given level of certain qualities (*nqual*) is to be mixed in a given number of pools (*npool*) in order to produce a number of products (*nprod*) with prespecified characteristics. The situation is described in Figure 7.6.

We use the reformulation proposed by (Ben-Tal *et al.*, 1994) in which one considers the fractional flow rates q_{il}. We address a generalized version of the

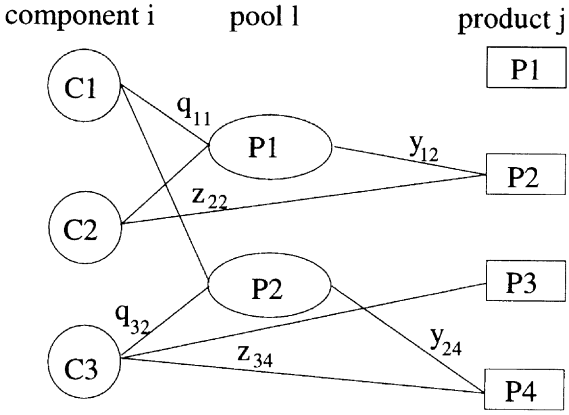


Figure 7.6. Pooling/Blending Problems

blending and pooling problem in which every component stream is allowed to directly reach any pool as well as any product. According to that assumption, the blending and pooling problem can be defined as follows:

$$\min_{q,x,z} - \sum_{j}^{nprod} \sum_{l}^{npool} (d_j - \sum_{i}^{ncomp} c_i q_{il}) y_{lj} - \sum_{j}^{nprod} \sum_{i}^{ncomp} (d_j - c_i) z_{ij}$$

s.t.

$$\begin{aligned}
\sum_{l}^{npool} \sum_{j}^{nprod} q_{il} y_{lj} + \sum_{j}^{nprod} z_{ij} &\leq A_i \ , \quad \forall i \\
\sum_{j}^{nprod} y_{lj} &\leq S_l \ , \quad \forall l \\
\sum_{l}^{npool} y_{lj} + \sum_{i}^{ncomp} z_{ij} &\leq D_j \ , \quad \forall j \\
\sum_{l}^{npool} (\sum_{i}^{ncomp} C_{ik} q_{il} - P_{jk}) y_{lj} + \sum_{i}^{ncomp} (C_{ik} - P_{jk}) z_{ij} &\leq 0 \ , \quad \forall j, k \\
\sum_{i}^{ncomp} q_{il} &= 1 \ , \quad \forall l \\
q_{il} \geq 0 \ , \quad y_{lj} \geq 0 \ , \quad z_{ij} &\geq 0.
\end{aligned}$$

The resulting mathematical model corresponds to a quadratically constrained problem with a quadratic objective function. Different instances of randomly generated Blending and Pooling Problems based on the above reformulation were generated and solved. Typical results of the distributed implementation of the GOP in an *Intel-Paragon* machine are shown in Table 7.5.

ncomp	nprod	npool	nqual	nvar	ncon	NI_c	PE	CPU(s)
5	5	3	5	55	186	15	64	15.3
10	4	4	9	96	300	15	64	36.3
10	4	4	16	96	356	15	64	42.0
10	4	4	18	96	372	16	64	39.0
10	4	4	25	96	428	16	64	44.9
10	4	5	30	110	468	20	64	843
12	4	4	9	112	336	16	64	37.3
12	4	4	25	112	464	16	64	50.1
12	4	4	30	112	504	16	64	44.9
12	4	5	4	128	330	20	64	869

Table 7.5. Blending and Pooling Problems

The total number of variables for each instance is $nvar = ncomp \times nprod + npool \times nprod + ncomp \times nprod$ and the total number of constraints $ncon = 2 \times ncomp + nprod + 2 \times nprod + 2 \times npool \times nqual + 2 \times nprod$. The distributed implementation of the GOP allowed us to address problems with 20 connected variables, which require the solution of 1,048,576 relaxed dual problems, in very realistic computational times. For instance, as can be seen from Table 7.5 pooling problems of 128 variables and 330 constraints can be solved within 840–870 s., even though they have 20 connected variables.

Summary

The focus on this chapter has been on the distributed implementation issues of the GOP global optimization approach. The updates of the bounds, the solution of the relaxed dual problems and the routing issues are discussed. The computational studies of the distributed implementation of the GOP in an *Intel-Paragon* machine, on large scale applications of pooling and blending problems, as well as indefinite quadratic programming problems demonstrate its potential.

II Signomial Problems

8 GENERALIZED GEOMETRIC PROGRAMMING : THEORY

In this chapter, we discuss the theoretical and algorithmic issues in global optimization approaches for generalized geometric programming problems. This material is based on the work of Maranas and Floudas (1997). Section 8.1 provides an introduction to generalized geometric programming. Section 8.2 focuses on the theoretical analysis for the development of a global optimization approach. Section 8.3 presents the global optimization algorithm and its proof of convergence to an ϵ-global optimum solution. Finally, section 8.4 presents an illustrative example.

8.1 INTRODUCTION

Generalized geometric programming *(GGP)* is the class of optimization problems where the objective function and constraints are *signomials*, that is they can be written as the difference of two *posynomials*. A *posynomial* $G(\boldsymbol{x})$ is simply the sum of a number of *posynomial terms* or *monomials* $g_k(\boldsymbol{x})$, $k = 1, \ldots, K$, multiplied by some positive real constants c_k, $k = 1, \ldots, K$:

$$G(\boldsymbol{x}) \;=\; c_1 g_1(\boldsymbol{x}) + c_2 g_2(\boldsymbol{x}) + \cdots + c_K g_K(\boldsymbol{x}). \tag{8.1}$$

Note that $c_k \in \Re^+$, $k = 1, \ldots, K$. Each monomial $g_k(\boldsymbol{x})$ is in turn the product of a number of positive variables each of them raised to some real power,

$$g_k(\boldsymbol{x}) \;=\; x_1^{d_{1,k}} x_2^{d_{2,k}} \cdots x_n^{d_{N,k}}, \;\; k = 1, \ldots, K \tag{8.2}$$

C. A. Floudas, *Deterministic Global Optimization*
© Springer Science+Business Media Dordrecht 2000

where $d_{1,k}, d_{2,k}, \ldots, d_{N,k} \in \Re$ and are not necessarily integer. The term geometric programming was adopted because of the key role that the well known arithmetic–geometric inequality played in the initial developments.

By grouping together monomials with identical sign, the generalized geometric programming *(GGP)* problem can be formulated as the following nonlinear optimization problem:

$$\begin{aligned}
\min_{\boldsymbol{t}} \quad & G_0(\boldsymbol{t}) = G_0^+(\boldsymbol{t}) - G_0^-(\boldsymbol{t}) \\
\text{subject to} \quad & G_j(\boldsymbol{t}) = G_j^+(\boldsymbol{t}) - G_j^-(\boldsymbol{t}) \leq 0, \quad j = 1, \ldots, M \quad (GGP) \\
& t_i \geq 0, \quad i = 1, \ldots, N \\
\text{where} \quad & G_j^+(\boldsymbol{t}) = \sum_{k \in K_j^+} c_{jk} \prod_{i=1}^{N} t_i^{\alpha_{ijk}}, \quad j = 0, \ldots, M \\
& G_j^-(\boldsymbol{t}) = \sum_{k \in K_j^-} c_{jk} \prod_{i=1}^{N} t_i^{\alpha_{ijk}}, \quad j = 0, \ldots, M
\end{aligned}$$

where $\boldsymbol{t} = (t_1, \ldots, t_N)$ is the positive variable vector; G_j^+, G_j^-, $j = 0, \ldots, M$ are positive posynomial functions in $\boldsymbol{t}$; α_{ijk} are arbitrary real constant exponents; whereas c_{jk} are given *positive* coefficients. Finally, sets K_j^+, K_j^- include the positively/negatively signed monomials that form posynomials G_j^+, G_j^- respectively. In general, formulation *(GGP)* corresponds to a highly nonlinear optimization problem with a nonconvex objective function and/or constraint set and possibly disjoint feasible region. Note that if we set $K_j^- = 0$ for all $j = 0, \ldots, M$ (i.e., we do not have negative monomials), then the mathematical model for *(GGP)* reduces to the (posynomial) geometric programming *(GP)* formulation:

$$\begin{aligned}
\min_{\boldsymbol{t}} \quad & G_0^+(\boldsymbol{t}) \\
\text{subject to} \quad & G_j^+(\boldsymbol{t}) \leq 0, \quad j = 1, \ldots, M \qquad (GP) \\
& t_i \geq 0, \quad i = 1, \ldots, N \\
\text{where} \quad & G_j^+(\boldsymbol{t}) = \sum_{k \in K_j^+} c_{jk} \prod_{i=1}^{N} t_i^{\alpha_{ijk}}, \quad j = 0, \ldots, M.
\end{aligned}$$

Posynomial geometric programming *(GP)* laid the foundation for the theory of generalized geometric programming *(GGP)* problems. Zener (1961,1971) along with Duffin (1962a,b) as well as Duffin and Peterson (1966), Duffin *et al.* (1967) pioneered the initial work on *(GP)*. The most important result is the derivation

of the exact (posynomial) geometric programming dual (Duffin *et al.*, 1967), based on the arithmetic–geometric inequality. This dual, after applying a logarithmic transformation on the objective function, involves the maximization of a separable concave function subject to linear constraints. Therefore, it has a single maximum which can be routinely found with available local nonlinear optimization algorithms (e.g., MINOS 5.4 (Murtagh and Saunders, 1983)). In fact, it can be shown that if there are exactly $N + 1$ monomials in the primal problem then the solution of the dual problem can be obtained by solving a system of linear equations. Since there is no duality gap (Duffin *et al.*, 1967), the single maximum of the dual problem matches the single minimum of *(GP)*.

Generalized geometric programming *(GGP)* problems were first introduced and studied by Passy and Wilde (1967a) and Blau and Wilde (1969) when existing (posynomial) geometric programming *(GP)* formulations failed to account for the presence of negatively signed monomials in models for important applications. A plethora of such applications exists in a very broad range of disciplines. Engineering design problems typically involve the minimization of a cost function or maximization of annual profit subject to a number of constraints, including steady state material and energy balances or transport equations (Blau and Wilde, 1969). Because most relations for cost functions or transport coefficients are derived from dimensional analysis or by fitting power functions to experimental data they are frequently posynomials. The breadth and depth of existing applications is indicated in an excellent bibliographic reference by Rijckaert and Martens (1978a), as well as the book edited by Avriel (1980). Chemical engineering applications include among others: heat exchanger network design (Duffin and Peterson, 1966); chemical reactor design (Blau and Wilde, 1969; Blau and Wilde, 1971); optimal condenser design (Avriel and Wilde, 1967); oxygen production (Hellinckx and Rijckaert, 1972); membrane separation process design (Dembo, 1978a); optimal design of cooling towers (Ecker and Wiebking, 1978); chemical equilibrium problems (Passy and Wilde, 1967b); optimal control (Jefferson and Scott, 1978); and batch plant modeling (Salomone and Iribarren, 1992; Hellinckx and Rijckaert, 1971).

Unlike (posynomial) *(GP)* problems, *(GGP)* problems remain nonconvex in both their primal and dual representation and no known transformation can convexify them. They involve multiple local minima and therefore are much more difficult problems to solve. A key development in the theory of *(GGP)* is the the concept of a ***condensed polynomial*** which is based on the classical arithmetic–geometric inequality which states that the weighted arithmetic mean of positive numbers $t_1, t_2, \ldots, t_N$ is greater than or equal to their geo-

metric mean:

$$\begin{aligned} \sum_{i=1}^{N} t_i &\geq \prod_{i=1}^{N} \left(\frac{t_i}{w_i}\right)^{w_i} \\ \text{where} \quad \sum_{i=1}^{N} w_i &= 1 \\ w_i &\geq 0, \quad i = 1, \ldots, N. \end{aligned} \tag{8.3}$$

Note that the equality holds if and only if,

$$\frac{t_1}{w_1} = \frac{t_2}{w_2} = \cdots = \frac{t_N}{w_N}.$$

By applying this inequality to a posynomial function we have,

$$\sum_{k \in K} c_k \prod_{i=1}^{N} t_i^{\alpha_{ik}} \geq \prod_{k \in K} \left[\frac{c_k \prod_{i=1}^{N} t_i^{\alpha_{ik}}}{w_k} \right]^{w_k}.$$

If $\bar{\boldsymbol{t}}$ is a given point and the weights $w_k(\boldsymbol{t})$ are selected as :

$$w_k(\bar{\boldsymbol{t}}) = \frac{c_k \prod_{i=1}^{N} \bar{t}_i^{\alpha_{ik}}}{\sum_{k \in K} c_k \prod_{i=1}^{N} \bar{t}_i^{\alpha_{ik}}}, \quad k \in K,$$

then the posynomial function $G(\boldsymbol{t})$ can be condensed into the monomial

$$g(\boldsymbol{t}) = \prod_{k \in K} \left[\prod_{i=1}^{N} \left(\frac{t_i}{\bar{t}_i}\right)^{\alpha_{ik}} G(\bar{\boldsymbol{t}}) \right]^{w_k(\bar{\boldsymbol{t}})} \leq \sum_{k \in K} c_k \prod_{i=1}^{N} t_i^{\alpha_{ik}} = G(\boldsymbol{t}).$$

Note that, $g(\boldsymbol{t}) \leq G(\boldsymbol{t})$ and only for $\boldsymbol{t} = \bar{\boldsymbol{t}}$, $g(\bar{\boldsymbol{t}}) = G(\bar{\boldsymbol{t}})$. Based on this condensation procedure, a number of algorithms for solving *(GGP)* have been proposed. Duffin (1970) introduced a reformulation of *(GGP)* involving constraints which are either convex or reverse convex (concave). By successively condensing the posynomials in the reverse convex constraints, an iterative procedure is defined for generating a sequence of posynomial problems whose minima are monotonically decreasing upper bounds on the global minimum of *(GGP)*. However, since *(GGP)* might have multiple minima this procedure is not guaranteed to converge to the global solution. A similar algorithm without using dual information is proposed by Duffin (1970), Avriel *et al.* (1975) where each inequality constraint is transformed into the ratio of two posynomials and the denominator is subsequently condensed into a monomial. Starting from a feasible primal point a series of *complementary problems* is constructed whose solution is shown to converge to an equilibrium point of the initial problem.

The arithmetic–harmonic mean inequality was used by Duffin and Peterson (1972) for solving signomial problems. Also based on the harmonic–arithmetic inequality Jefferson and Scott (1978) maximized the dual of the condensed problem based on a modified Newton method. Blau and Wilde (1971) and Kuester and Mize (1973) proposed an iterative procedure for solving the KKT conditions for the dual problem using the separability of the logarithmic equations. Also, Rijckaert and Martens (1978b) successively solved the KKT necessary optimality conditions, after condensing them into monomials with the geometric–arithmetic inequality and linearizing them with a logarithmic transformation.

Adaptations of general purpose nonlinear programming methods have been extensively used to solve signomial problems including the generalized reduced gradient (GRG) method (Ratner *et al.*, 1978), linear approximation method (Rijckaert and Martens, 1978b), separable programming (Kochenberger *et al.*, 1973), projection and restriction methods (Abrams and Wu, 1978), quasi Newton–Raphson type methods (Bradley, 1973), concave simplex method (Beck and Ecker, 1975), constrained Fletcher and Powell method (Haarhoff and Buys, 1970). A computational comparison of available codes for signomial programming is given by Dembo (1978b), and Rijckaert and Martens (1978b).

The main limitation of the aforementioned algorithms is that they can guarantee convergence to local minima at best. This represents an important limitation for *(GGP)* problems which are typically characterized by the presence of many local minima and disconnected feasible regions.

There has been little work on applying specialized global optimization algorithms with deterministic ϵ–convergence guarantee to locating the global minimum of problem *(GGP)*. Falk (1973) proposed such an algorithm for globally solving *(GGP)* based on convex relaxation and branch and bounding on the space of exponents of negative monomials ($j = 1, \ldots, M$ and $k \in K_j^-$). Maranas and Floudas (1997) proposed a deterministic global optimization approach for *(GGP)* problems which is based on (i) a difference of convex functions' transformation, (ii) convex lower bounding, (iii) variable scaling, (iv) monotonicity analysis, and (v) a branch and bound framework. Sherali (1998) proposed a global optimization approach for polynomial programming problems with rational exponents based on the reformulation-linearization framework of Sherali and Alameddine (1992) and Sherali and Tuncbilek (1992).

8.2 THEORETICAL ANALYSIS

In this section, we study and analyze the mathematical structure of generalized geometric programming *(GGP)* problems.

8.2.1 Difference of two Convex Functions Transformation

The objective function and the constraints in the original formulation *GGP* are in general nonconvex functions. By applying the transformation,

$$t_i = \exp z_i, \quad i = 1, \ldots, N$$

on the original formulation *(GGP)* we obtain the following programming problem *(DC)*.

$$\begin{aligned}
\min_{z} \quad & G_0(z) = G_0^+(z) - G_0^-(z) \\
\text{subject to} \quad & G_j(z) = G_j^+(z) - G_j^-(z) \le 0, \quad j = 1, \dots, M \qquad (DC) \\
& z_i^L \le z_i \le z_i^U, \quad i = 1, \dots, N \\
\text{where} \quad & G_j^+(z) = \sum_{k \in K_j^+} c_{jk} \exp\left\{ \sum_{i=1}^{N} \alpha_{ijk}\, z_i \right\}, \quad j = 0, \dots, M \\
& G_j^-(z) = \sum_{k \in K_j^-} c_{jk} \exp\left\{ \sum_{i=1}^{N} \alpha_{ijk}\, z_i \right\}, \quad j = 0, \dots, M.
\end{aligned}$$

Note that because $z_i^L = \log(t_i^L)$, $z_i^U = \log(t_i^U)$ it is necessary that the lower bounds t_i^L of all t_i must be strictly positive for such a reformulation to exist. This problem, however, can be overcome by appropriately scaling all problematic original variables t_i so that

$$t_i' = t_i + \max\{0, -t_i^L + \epsilon\}, \quad \epsilon > 0.$$

Because the function $(\exp x)$ is convex, the exponent of a linear expression, (i.e., $\exp\left\{ \sum_{i=1}^{N} \alpha_{ijk}\, z_i \right\}$) is also convex. Furthermore, all functions $G_j^+(z), G_j^-(z)$ that are positive linear combinations of convex functions are convex as well. This means that both objective function and nonlinear constraints in *(DC)* can be viewed as the difference of two convex functions. Note that if $G_j^-(z) = 0$ for every $j = 0, \dots, M$ then problem *(DC)* becomes a convex programming problem. One could further transform problem *(DC)* by defining the additional variables,

$$Y_{jk} = \sum_{i=1}^{N} \alpha_{ijk} z_i, \quad j = 0, \dots, M \text{ and } k \in K_j^+ \cup K_j^-,$$

into a programming problem involving *separable* objective function and nonlinear constraints plus a set of linear equality constraints. However, this greatly increases the total number of nonlinear variables. Therefore, in this work we implicitly take advantage of this separability without explicitly defining any new variables.

8.2.2 Convex Lower Bounding

A lower bound on the solution of problem *(DC)* can be obtained by solving a convex relaxation of the original problem *(DC)*. Such a convex relaxation can

be realized by underestimating every concave function, $-G_j^-(z)$, with a linear function $-L_j^-(z)$ for every $j = 0, \ldots, M$. This linear function is constructed by underestimating every implicitly separable term $-\exp\left\{\sum_{i=1}^{N} \alpha_{ijk}\, z_i\right\}$ of $-G_j^-(z)$ with a linear function. This defines the following relaxed convex programming problem *(R)* whose solution provides a lower bound on the solution of *(DC)*.

$$
\begin{aligned}
\min_{z} \quad & G_0^{conv}(z) = G_0^+(z) - L_0^-(z) \\
\text{subject to} \quad & G_j^{conv}(z) = G_j^+(z) - L_j^-(z) \le 0, \quad j = 1, \ldots, M \qquad (R) \\
& z_i^L \le z_i \le z_i^U, \quad i = 1, \ldots, N \\
\text{where} \quad & G_j^+(z) = \sum_{k \in K_j^+} c_{jk} \exp\left\{\sum_{i=1}^{N} \alpha_{ijk} z_i\right\}, \; j = 0, \ldots, M \\
& L_j^-(z) = \sum_{k \in K_j^-} c_{jk} \left\{A_{jk} + B_{jk}\left(\sum_{i=1}^{N} \alpha_{ijk} z_i\right)\right\}, \\
& \qquad j = 0, \ldots, M, \quad k \in K_j^- \\
\text{and} \quad & A_{jk} = \frac{Y_{jk}^U \exp(Y_{jk}^L) - Y_{jk}^L \exp(Y_{jk}^U)}{Y_{jk}^U - Y_{jk}^L} \\
& B_{jk} = \frac{\exp(Y_{jk}^U) - \exp(Y_{jk}^L)}{Y_{jk}^U - Y_{jk}^L}, \\
& Y_{jk}^L = \sum_{i=1}^{N} \min\left(\alpha_{ijk} z_i^L, \alpha_{ijk} z_i^U\right) \\
& Y_{jk}^U = \sum_{i=1}^{N} \max\left(\alpha_{ijk} z_i^L, \alpha_{ijk} z_i^U\right).
\end{aligned}
$$

Note that the linear underestimator $-L_j^-(z)$ of $-G_j^-(z)$ consists of the sum of a number of linear functions each one of which is lower bounding an implicitly univariate function of the form $-\exp(Y)$. Clearly, the smaller the difference between the original functions $G_j^-(z)$ and the linearizations $L_j^-(z)$ the closer the solution of *(R)* will be to the solution of *(DC)*. The quality of this lower bounding can be analyzed by examining the tightness of underestimation of every concave term of the form $-\exp(Y)$ with a linear function inside some interval $[Y^L, Y^U]$. Let $\Delta(Y)$ be the separation between the concave function $-\exp(Y)$ and the linear underestimator inside the interval $[Y^L, Y^U]$ (See Figure 8.1).

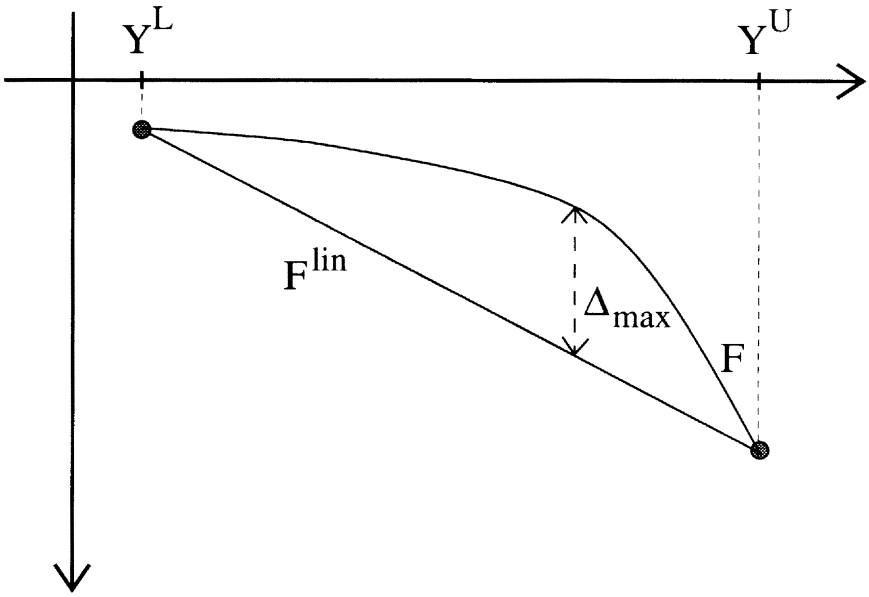


Figure 8.1. Linear underestimation of $-\exp(Y)$

$$\Delta(Y) = -\exp(Y) - \left(\frac{Y^U \exp(Y^L) - Y^L \exp(Y^U)}{Y^U - Y^L} + \frac{\exp(Y^U) - \exp(Y^L)}{Y^U - Y^L} Y\right).$$

This separation function $\Delta(Y)$ is concave in Y and it reaches its single maximum at

$$Y^* = \log\left(\frac{\exp(Y_{jk}^U) - \exp(Y_{jk}^L)}{Y_{jk}^U - Y_{jk}^L}\right)$$

with a value

$$\begin{aligned} \Delta_{max} &= \exp(Y^L)(1 - Z + Z\log(Z)) \\ \text{where} \quad Z &= \frac{\exp(\delta) - 1}{\delta}, \quad \delta = Y^U - Y^L. \end{aligned}$$

Note that as the interval width $\delta = Y^U - Y^L$ goes to zero, Z approaches one and therefore the maximum separation goes to zero.

$$\delta \longrightarrow 0, \quad Z \longrightarrow 1, \text{ and } \Delta_{max} \longrightarrow 0.$$

$\delta = Y^U - Y^L$	$\Delta_{max}/\exp(Y^L)$
0.00026	10^{-8}
0.00275	10^{-6}
0.02808	10^{-4}
0.26449	10^{-2}
1.75079	1
2	1.51572
5	71.2807
10	14752.3

Table 8.1. Maximum separation as a function of interval width

The rate at which this maximum separation goes to zero can be determined by the Taylor expansion of $\Delta_{max}(\delta)$ at $\delta = 0$.

$$\frac{\Delta_{max}}{\exp(Y^L)} = \frac{\delta^2}{8} + \frac{\delta^3}{16} + \frac{11\delta^4}{576} + \frac{5\delta^5}{1152} + \frac{41\delta^6}{51840} + \frac{5\delta^7}{41472} + \mathcal{O}(\delta^8).$$

By considering only the first term of the positive termed series expansion we deduce that the rate at which Δ_{max} approaches zero as δ goes to zero is

$$\Delta_{max} \approx \mathcal{O}(\delta^2), \text{ as } \delta \longrightarrow 0.$$

On the other hand, as δ goes to infinity Δ_{max} goes to infinity as well. In this case, however, Δ_{max} goes to infinity as

$$\Delta_{max} \approx \mathcal{O}(\exp(\delta)), \text{ as } \delta \longrightarrow +\infty.$$

The scaled maximum separation $\Delta_{max}/\exp(Y^L)$ for different values of δ is shown in Table 8.1. Note that for small values of $\delta < \approx 0.5$ the maximum separation, Δ_{max}, goes as $\mathcal{O}(\delta^2)$ following the theoretically derived predictions. If for example, a maximum separation of 10^{-4} is required, an interval width of just 10^{-2} will suffice. On the other hand, for larger values of $\delta > \approx 2$ the maximum separation grows exponentially with δ rendering this underestimating scheme inefficient. This implies that the difference between the upper and lower bounds of all $\sum_{i=1}^{N} \alpha_{ijk} z_i$ linear terms for every $j = 1, \ldots, M$ and every $k \in K_j^-$ must be as small as possible if an effective lower bounding is desired.

8.2.3 Variable Scaling

The variable scaling can be accomplished by first scaling all variables t_i in the original formulation *(GGP)* and then employing the exponential transformation. *This variable scaling can be accomplished if all α_{ijk}'s are integrals or rational numbers that can be reduced into integrals.* Note that, by preserving all constants in the transformed objective function and constraints, the same convergence and feasibility tolerances can be maintained. Such a scaling is the following

$$t_i \longleftarrow t_i^L + \frac{t_i^U - t_i^L}{t_i^{U,new} - t_i^{L,new}} \left(t_i^{new} - t_i^{L,new}\right), \quad i = 1, \ldots, N$$

where $t_i^{L,new}, t_i^{U,new}$ are selected so as $\log(t_i^{U,new}) - \log(t_i^{L,new})$ is small. The effect of this scaling on the maximum separation can be realized by considering the following simple example :

$$\begin{aligned} &\min -t \\ \text{subject to} \quad 0 < t^L = \exp(Y^L) \leq \; & t \; \leq \exp(Y^U) = t^U. \end{aligned}$$

It was shown earlier that the maximum separation between the transformed concave objective function and the linear underestimation is

$$\Delta_{max} = \exp(Y^L)\left[1 - \frac{\exp(\delta) - 1}{\delta} + \frac{\exp(\delta) - 1}{\delta} \log \frac{\exp(\delta) - 1}{\delta}\right].$$

The single variable t of the problem can be scaled as follows:

$$t \longleftarrow t^L + \frac{t^U - t^L}{t^{U,new} - t^{L,new}} \left(t^{new} - t^{L,new}\right).$$

Without loss of generality we select $t^{L,new} = \exp(Y^{L,new}) = 1$ which means that $Y^{L,new} = 0$ and therefore $Y^{U,new} = \delta^{new}$. The original minimization problem now becomes,

$$\begin{aligned} \min \quad & -\left[\exp(Y^L) + \frac{\exp(Y^U) - \exp(Y^L)}{\exp(\delta^{new}) - 1}(t^{new} - 1)\right] \\ \text{subject to} \quad & 1 \leq t \leq \exp(\delta^{new}). \end{aligned}$$

The maximum separation after rescaling is now

$$\begin{aligned} \Delta_{max}^{new} &= \exp(Y^L)\frac{\exp(\delta) - 1}{\exp(\delta^{new}) - 1}\left[1 - \frac{\exp(\delta^{new}) - 1}{\delta^{new}}\right] \\ &+ \exp(Y^L)\frac{\exp(\delta) - 1}{\exp(\delta^{new}) - 1}\left[\frac{\exp(\delta^{new}) - 1}{\delta^{new}} \log \frac{\exp(\delta^{new}) - 1}{\delta^{new}}\right]. \end{aligned}$$

The ratio of the maximum separation for the scaled example over the maximum separation for the original one is therefore,

$$R(\delta, \delta^{new}) = \frac{\Delta_{max}^{new}}{\Delta_{max}} = \left\{ \frac{\frac{1}{\exp(\delta^{new})-1} - \frac{1}{\delta^{new}} + \frac{1}{\delta^{new}} \log \frac{\exp(\delta^{new})-1}{\delta^{new}}}{\frac{1}{\exp(\delta)-1} - \frac{1}{\delta} + \frac{1}{\delta} \log \frac{\exp(\delta)-1}{\delta}} \right\}.$$

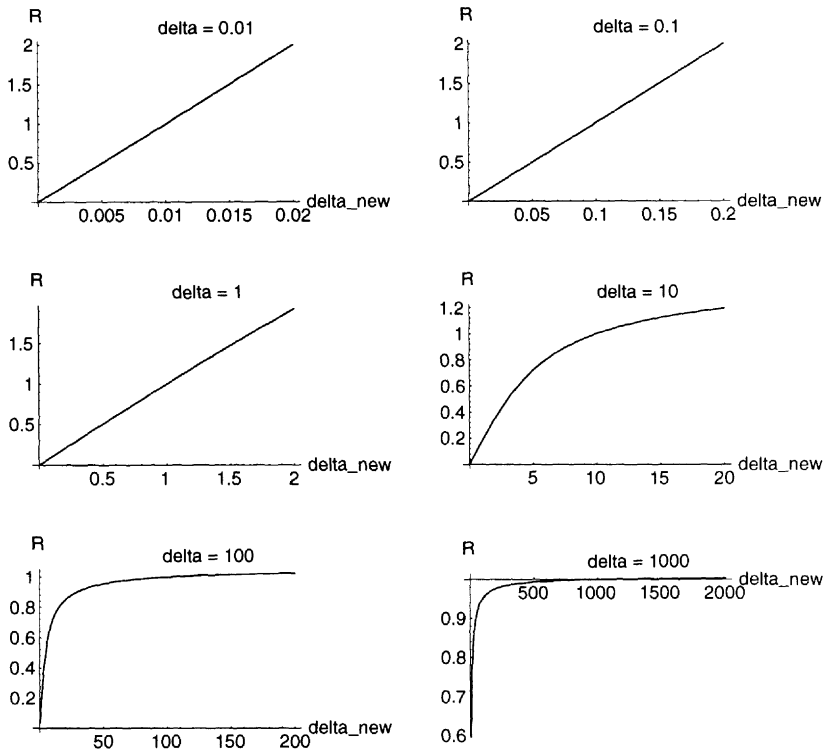


Figure 8.2. Ratio R of maximum separations

The effect of this rescaling on the quality of the underestimation can be evaluated by plotting the ratio R as a function of the arbitrarily selected new interval width δ^{new} for different values of the initial interval width $\delta = 0.01, 0.1, 1, 10, 100$ (see Figure 8.2). From Figure 8.2 it is clear that we can always improve on the initial underestimation $R \leq 1$ by choosing $\delta^{new} \leq \delta$. This implies that one can improve on the underestimation of a negative exponential of a linear function by rescaling the argument so that the difference between its bounds is small. In fact it can be shown that

$$\lim_{\delta^{new} \to 0} R(\delta, \delta^{new}) = 0$$

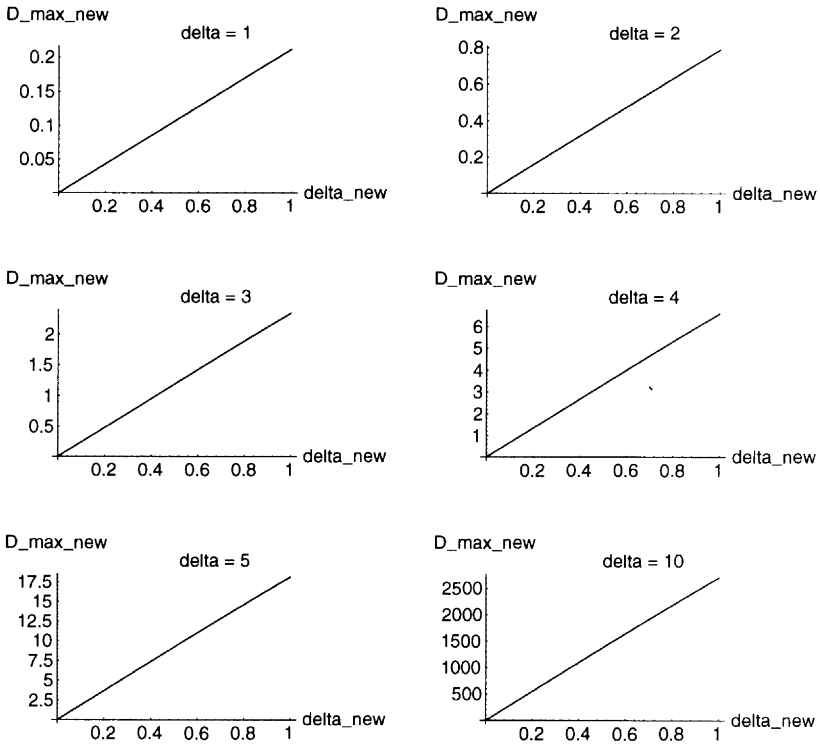


Figure 8.3. Maximum separation Δ_{max}^{new} of scaled example

in practice, however, round–off errors do not allow for δ^{new} to become less than ≈ 0.01. By plotting Δ_{max}^{new} as a function of δ^{new} for different values of $\delta = 1, 2, 3, 4, 5, 10$ (see Figure 8.3) we observe that the larger the value of the initial δ the smaller the value of δ_{new} should be. It must be emphasized, however, that *the proposed variable scaling does not affect the scaling of the constraints.*

This implies that the same feasibility tolerance ϵ_f can still be utilized before and after the variable scaling. The effect of this rescaling procedure on the global optimization algorithm is shown in the illustrative example.

8.2.4 Transformation of Inequalities

The feasible region of the original problem *(GGP)* is the intersection of the nonlinear inequality constraints

$$G_j(\boldsymbol{t}) \;=\; G_j^+(\boldsymbol{t}) - G_j^-(\boldsymbol{t}) \;\leq\; 0, \;\; j = 1, \ldots, M$$

which lies within the hyperectangle defined by the box constraints

$$0 \;<\; t_i^L \;\leq\; t_i \;\leq\; t_i^U, \;\; i = 1, \ldots, N.$$

The convex relaxation *(R)* of *(GGP)* approximates the original feasible region with a relaxation of its convex hull. The closer this relaxation is to its convex hull the tighter the lower bounding of the objective function will be. This motivates the need to preprocess the original nonlinear constraints so as the employed relaxation will be as tight as possible. This can be accomplished by taking advantage of the mathematical structure of each original constraint $G_j(\boldsymbol{t})$ and the fact that all variables $\boldsymbol{t}$ are strictly positive. The positivity of $\boldsymbol{t}$ implies that for every nonlinear inequality constraint $G_j(\boldsymbol{t})$ there exists a family of nonlinear constraints $\mathcal{G}_j(\boldsymbol{t})$ whose elements are completely interchangeable with $G_j(\boldsymbol{t})$ and are derived by multiplying the original constraint by $t_1^{d_{1j}} t_2^{d_{2j}} \cdots t_N^{d_{Nj}}$, where d_{ij}, $i = 1, \ldots, N$, $j = 1, \ldots, M$ are arbitrarily selected real parameters :

$$\mathcal{G}_j(\boldsymbol{t}) \;=\; \left\{ G_j(\boldsymbol{t}, \boldsymbol{d}_j) \;=\; G_j(\boldsymbol{t}) \left(\prod_{i=1}^{N} t_i^{d_{ij}} \right) \;:\; d_{ij} \;\in\; \Re, \; i = 1, \ldots, N \right\}.$$

Note that even though the values of the elements of $\mathcal{G}_j$ might not be identical for every $\boldsymbol{t}$, because $t_1^{d_{1j}}, \ldots, t_N^{d_{Nj}} \;>\; 0$, they always maintain the same sign as $G_j(\boldsymbol{t})$. This implies that every element of $\mathcal{G}_j(\boldsymbol{t})$

$$G_j(\boldsymbol{t}, \boldsymbol{d}_j) \;=\; \sum_{k \in K_j^+} c_{jk} \prod_{i=1}^{N} t_i^{(\alpha_{ijk} + d_{ij})} \;-\; \sum_{k \in K_j^-} c_{jk} \prod_{i=1}^{N} t_i^{(\alpha_{ijk} + d_{ij})}$$

is an equivalent representation of the original constraint and therefore can be used to replace the original constraints in formulation *(GGP)*. Clearly, unlike the variable scaling presented in the previous subsection, this multiplicative transformation affects the scaling of the problem. Therefore, appropriate changes have to be made on the feasibility as well as convergence tolerances to account for the changed scaling of the objective function and constraints.

After applying the exponential transformation we obtain :

$$\begin{aligned} G_j(\boldsymbol{z}, \boldsymbol{d}_j) &= \sum_{k \in K_j^+} c_{jk} \exp\left[\sum_{i=1}^{N} (\alpha_{ijk} + d_{ij})\, z_i\right] \\ &- \sum_{k \in K_j^-} c_{jk} \exp\left[\sum_{i=1}^{N} (\alpha_{ijk} + d_{ij})\, z_i\right], \end{aligned}$$

which can replace the constraint $G_j(\boldsymbol{z})$ in problem *(DC)*. Even though all constraints $G_j(\boldsymbol{z}, \boldsymbol{d}_j) \in \mathcal{G}_j$ are interchangeable, this is not the case for their convex relaxations. By linearizing the concave terms $G_j^-(\boldsymbol{z}, \boldsymbol{d}_j)$, different elements of $\mathcal{G}_j$ give rise to distinctly different convex relaxed constraints $G_j^{conv}(\boldsymbol{z}, \boldsymbol{d}_j)$ whose domain of feasibility depends on the selection of $\boldsymbol{d}_j$:

$$\begin{aligned} G_j^{conv}(\boldsymbol{z}, \boldsymbol{d}_j) &= \sum_{k \in K_j^+} c_{jk} \exp\left[\sum_{i=1}^{N} (\alpha_{ijk} + d_{ij})\, z_i\right] \\ &- \sum_{k \in K_j^-} c_{jk} \left\{ A_{jk} + B_{jk} \left[\sum_{i=1}^{N} (\alpha_{ijk} + d_{ij})\, z_i\right] \right\} \end{aligned}$$

where

$$\begin{aligned} A_{jk} &= \frac{Y_{jk}^U \exp(Y_{jk}^L) - Y_{jk}^L \exp(Y_{jk}^U)}{Y_{jk}^U - Y_{jk}^L} \\ B_{jk} &= \frac{\exp(Y_{jk}^U) - \exp(Y_{jk}^L)}{Y_{jk}^U - Y_{jk}^L} \\ Y_{jk}^L &= \sum_{i=1}^{N} \min\left[(\alpha_{ijk} + d_{ij})\, z_i^L, (\alpha_{ijk} + d_{ij})\, z_i^U\right] \\ Y_{jk}^U &= \sum_{i=1}^{N} \max\left[(\alpha_{ijk} + d_{ij})\, z_i^L, (\alpha_{ijk} + d_{ij})\, z_i^U\right]. \end{aligned}$$

Because the tightest lower bounding of $G_j(\boldsymbol{z}, \boldsymbol{d}_j)$ is sought, for a given $\boldsymbol{z}$ the best selection of $\boldsymbol{d}_j$ is the one for which $G_j(\boldsymbol{z}, \boldsymbol{d}_j)$ is maximized,

$$\boldsymbol{d}_j^*(\boldsymbol{z}) = \arg\max_{\boldsymbol{d}_j} G_j(\boldsymbol{z}, \boldsymbol{d}_j).$$

Note that, it is not possible to consider the convex relaxations of all elements of $\mathcal{G}_j$ because their number is infinite. Unfortunately, the maximization problem is not only parametric in $\boldsymbol{z}$ but also, for a given $\boldsymbol{z}$, is as difficult to solve as the original problem *(DC)*. Therefore, instead of solving it a number of criteria can be utilized to guide the selection of a good value for $\boldsymbol{d}_j$ for every constraint j.

The first criterion is based on the objective to minimize the number of variables z_i participating in every concave function $G_j^-(\boldsymbol{z}, \boldsymbol{d}_j)$. This requirement reflects the primary concern of any deterministic global optimization algorithm

which is to keep the number of variables where branching is required at a minimum. By utilizing the unit function δ_o,

$$\delta_o(x) = \begin{cases} 1 & \text{for } x = 0 \\ 0 & \text{otherwise} \end{cases}$$

this criterion can be expressed mathematically for every constraint j as the following problem (C1):

$$\min_{d_{ij}} \quad \sum_{i=1}^{N} \left[1 - \delta_o \left(\sum_{k \in K_j^-} |\alpha_{ijk} + d_{ij}| \right) \right], \quad j = 1, \ldots, M. \tag{C1}$$

Note that if, for a variable i in a given constraint j there exists $\boldsymbol{d}_j$ such that

$$\sum_{k \in K_j^-} |\alpha_{ijk} + d_{ij}| = 0$$

then variable i can be eliminated from the concave part of constraint j. The solution of (C1) provides the corresponding values of $\boldsymbol{d}_j$ for which there exists a minimum number of variables, z_i, participating in $G_j^-(\boldsymbol{z}, \boldsymbol{d}_j)$ and identifies these participating variables, z_i, to be the ones for which,

$$\sum_{k \in K_j^-} |\alpha_{ijk} + d_{ij}| > 0.$$

Note that (C1) might have multiple solutions. In this case more than one or even all the corresponding convex relaxed constraints can be incorporated in the *(R)* formulation.

The second criterion is associated with the observation that for a given constraint j, the selection of $\boldsymbol{d}_j$ aims at maintaining $Y_{jk}^U - Y_{jk}^L$ as small as possible for every $k \in K_j^-$. Although the rescaling of variables aims at this, sometimes it is necessary to complement this rescaling with an appropriate selection of $\boldsymbol{d}_j$. Because the primary concern is not to allow any $Y_{jk}^U - Y_{jk}^L$ to become larger than $\approx 2-5$, rather than minimizing a weighted sum of intervals $Y_{jk}^U - Y_{jk}^L$, it appears to be more meaningful to minimize the maximum $Y_{jk}^U - Y_{jk}^L$ over all $k \in K_j^-$ (Worst case analysis). This can be formulated as the following optimization problem (C2):

$$\min_{d_{ij}} \; Y_j^{max}, \quad j = 1, \ldots, M \tag{C2}$$

$$\begin{aligned} \text{subject to} \quad Y_j^{max} \geq & \sum_{i=1}^{N} \max\left[(\alpha_{ijk} + d_{ij})\, z_i^L, (\alpha_{ijk} + d_{ij})\, z_i^U\right] \\ & - \sum_{i=1}^{N} \min\left[(\alpha_{ijk} + d_{ij})\, z_i^L, (\alpha_{ijk} + d_{ij})\, z_i^U\right], \quad \forall\, k \in K_j^-. \end{aligned}$$

Apart from the aforementioned criteria for selecting $\boldsymbol{d}_j$ for a given constraint j, a number of heuristic rules can be utilized by exploiting the structure of specific problems. Furthermore, more than one convex relaxed function $G_j^{conv}(\boldsymbol{z}, \boldsymbol{d}_j)$ per constraint j can be incorporated into formulation (R) to tighten the approximation of the convex hull of constraint $G_j(\boldsymbol{z})$. Although no rigorous way is provided for selecting the optimal set of $\boldsymbol{d}_j$ per constraint, computational experience reveals that by utilizing (C1) and (C2), efficient convex underestimation of every constraint j can be achieved.

8.2.5 *Reduction of Rectangular Partitions*

The convex relaxation *(R)* of the original problem *(DC)* is aimed at deriving a lower bound on the solution of *(DC)*. This lower bound is calculated by solving the convex programming problem *(R)* inside some hyper-rectangle defined by:

$$z_i^L \leq z_i \leq z_i^U, \quad i = 1, \ldots, N.$$

Clearly, the smaller this hyper-rectangle is, the tighter the convex lower bounding of all $G_j(\boldsymbol{z})$, $j = 0, \ldots, M$ will be and therefore the closer the solution of *(R)* will be to the solution of *(DC)*. Note that we can not arbitrarily eliminate parts of the hyper-rectangle because we might exclude the global minimum solution. Therefore, deterministic global optimization algorithms improve the quality of the obtained lower bounds by partitioning the initial hyper-rectangle into smaller ones, and repeatedly solving *(R)* inside each one of these regions. A number of branching rules are utilized which make the subdivision process as efficient as possible.

To reduce the search domain, one can locate the minimum and maximum value for every variable z_i for which the nonlinear constraint set of *(DC)* remains feasible, while also imposing the constraint that the value of the objective function is at least as good as the best current upper bound U:

$$\begin{array}{ll} \min / \max & z_{i'}, \quad i' = 1, \ldots, N \\ \text{subject to} & G_j(\boldsymbol{z}) \leq 0, \quad j = 1, \ldots, M \\ & G_0(\boldsymbol{z}) \leq U \\ & z_i^L \leq z_i \leq z_i^U, \quad i = 1, \ldots, N. \end{array}$$

Since this is a nonconvex problem one can solve a relaxation of this problem by replacing all nonlinear constraints with their respective convex relaxations:

$$\begin{array}{ll} \min & z_{i'}, \quad i' = 1, \ldots, N \\ \text{subject to} & G_j^{conv}(\boldsymbol{z}) \leq 0, \quad j = 1, \ldots, M \\ & G_0^{conv}(\boldsymbol{z}) \leq U \\ & z_i^L \leq z_i \leq z_i^U, \quad i = 1, \ldots, N. \end{array} \qquad (BR)$$

The solutions of these convex bound reducing *(BR)* problems provide valid lower/upper bounds on all z_i so that all relaxed nonlinear constraints are feasi-

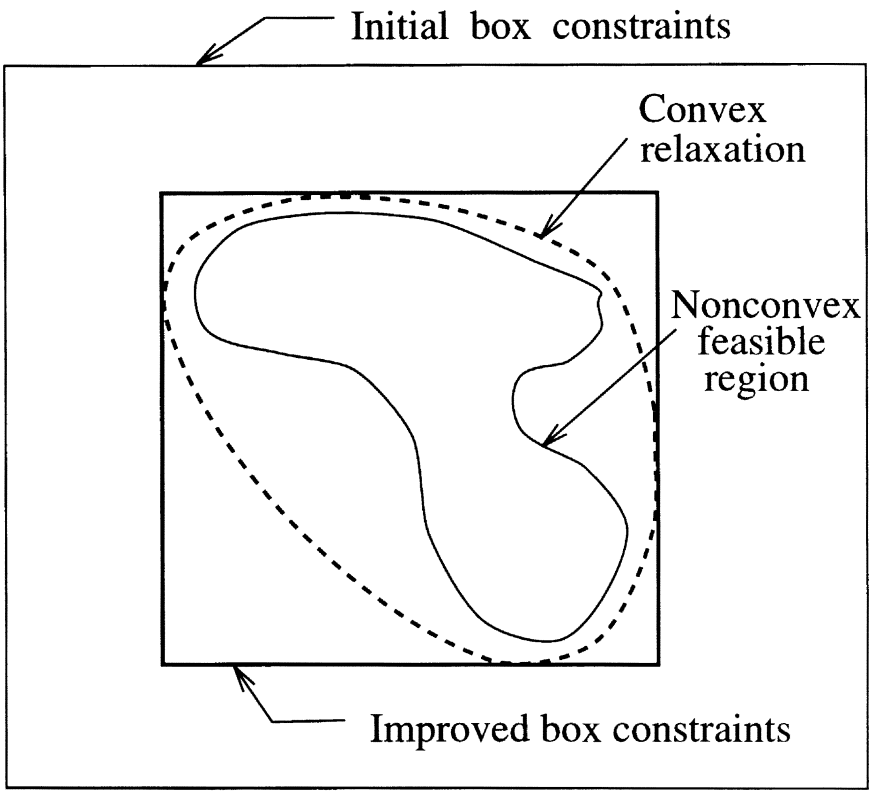


Figure 8.4. Initial bounds and feasible region, convex relaxation and improved bounds

ble and the relaxed objective function is less than the current best upper bound U. Note that if *(BR)* is infeasible then the corresponding rectangular partition can be eliminated because the global solution is guaranteed to be situated outside of it. As shown in Figure 8.4 by minimizing/maximizing all z_i, the feasible region of *(R)* is inscribed inside the smallest possible rectangle defining a new refined set of bounds for z. Because the reduction of the set of box constraints results in tighter underestimating functions $G_j^{conv}(z), G_0^{conv}(z)$ the minimization/maximization of all z_i can be repeated by dynamically updating the bounds z_i^L, z_i^U for more than one iteration until no further significant improvement on the bounds is achieved. It must be noted, however, that $2N$ convex programming problems must be solved per iteration. For computational efficiency reasons, this procedure can be limited to the initial partition. For intermediate rectangular partitions a relaxation of the proposed procedure is

utilized for detecting whether the current partition element can be fathomed. This alternative check is based on first deriving a lower bound for the objective function and every constraint of formulation *(R)* using monotonicity analysis principles. If the lower bound of the objective function is greater than U or if the lower bound of any of the constraints is greater than zero then the corresponding partition element can be eliminated since it is guaranteed either to not contain the global minimum solution or to be infeasible. The form of the objective and constraints in formulation *(R)* is

$$\begin{aligned} G_j^{conv}(\boldsymbol{z}) &= \sum_{k \in K_j^+} c_{jk} \exp\left(\sum_{i=1}^{N} Y_{jk}\right) \\ &- \sum_{k \in K_j^-} c_{jk} \left[A_{jk} + B_{jk}\left(\sum_{i=1}^{N} Y_{jk}\right)\right], \quad j = 0, \ldots, M \\ \text{where} \quad A_{jk} &= \frac{Y_{jk}^U \exp(Y_{jk}^L) - Y_{jk}^L \exp(Y_{jk}^U)}{Y_{jk}^U - Y_{jk}^L} \\ B_{jk} &= \frac{\exp(Y_{jk}^U) - \exp(Y_{jk}^L)}{Y_{jk}^U - Y_{jk}^L} \geq 0 \\ Y_{jk} &= \sum_{i=1}^{N} \alpha_{ijk} z_i, \; Y_{jk}^L = \sum_{i=1}^{N} \min\left(\alpha_{ijk} z_i^L, \alpha_{ijk} z_i^U\right) \\ Y_{jk}^U &= \sum_{i=1}^{N} \max\left(\alpha_{ijk} z_i^L, \alpha_{ijk} z_i^U\right). \end{aligned}$$

Since the function $\exp(x)$ is monotonically increasing, a lower bound $G_j^{conv,L}$ of $G_j^{conv}(\boldsymbol{z})$ can be obtained by fixing all Y_{jk}, $k \in K_j^+$ at their lower bounds Y_{jk}^L, and all Y_{jk}, $k \in K_j^-$ at their respective upper bounds Y_{jk}^U since $B_{jk} \geq 0$:

$$\begin{aligned} G_j^{conv,L} &= \sum_{k \in K_j^+} c_{jk} \exp\left(\sum_{i=1}^{N} Y_{jk}^L\right) \\ &- \sum_{k \in K_j^-} c_{jk} \left[A_{jk} + B_{jk}\left(\sum_{i=1}^{N} Y_{jk}^U\right)\right], \quad j = 0, \ldots, M. \end{aligned}$$

The corresponding partition element can be eliminated if

$$G_0^{conv,L} \geq U, \text{ or } \exists\, j = 1, \ldots, M \text{ such that } G_j^{conv,L} \geq 0.$$

These feasibility checks can be applied to every subdivision element before the convex minimization of problem *(R)* is performed, since the associated computational effort is negligible. Computational experience has shown that this results in significant reduction in the number of required convex minimizations (10% – 50%).

8.2.6 Monotonicity Analysis

Monotonicity analysis principles can be utilized to improve the convex relaxation of *(DC)* by dictating that some nonlinear inequality constraints must be satisfied as equality constraints at the global minimum solution. This results in a reduction of the degrees of freedom for problem *(DC)* through elimination of variables or introduction of equality constraints. Hansen *et al.* (1989) pointed out that if there exists a global minimum solution for *(DC)* and the objective function $G_0(\boldsymbol{z})$ is monotonically increasing(decreasing) in some z_i, then at least one constraint which is monotonically decreasing(increasing) with respect to z_i will be active at the global minimum solution. Note also that if z_i is not present in the objective function then at least one constraint involving z_i will be active at the global minimum solution. The conclusions from the monotonicity analysis can be more conveniently expressed by associating a boolean variable a_j for every constraint $j = 1, \ldots, M$. If the constraint j is active at the global minimum solution then $a_j = 1$, otherwise $a_j = 0$. Let also J_i^-, J_i^+, J_i^0 denote the sets of indices of constraints j involving variables z_i which are monotonically decreasing, monotonically increasing, and not monotonic or with unknown monotonicity in z_i, respectively. The monotonicity principles can then be expressed mathematically as follows:

$$\sum_{j \in J_i^+ \cup J_i^0} a_j \geq 1, \quad \text{if } G_0(\boldsymbol{z}) \text{ is monotonically decreasing in } z_i$$

$$\sum_{j \in J_i^- \cup J_i^0} a_j \geq 1, \quad \text{if } G_0(\boldsymbol{z}) \text{ is monotonically increasing in } z_i$$

$$\sum_{j \in J_i^+ \cup J_i^0} a_j \geq 1 \text{ or } \sum_{j \in J_i^- \cup J_i^0} a_j \geq 1, \quad \text{if } G_0(\boldsymbol{z}) \text{ is independent of } z_i.$$

Clearly, the smaller the sets J_i^0 are, the fewer a_j's will participate in any inequality relations, and therefore the more likely it will be to have some a_j's fixed at one. Note that if $a_j = 1$, then the corresponding constraint j is satisfied as an equality. This can lead to elimination of a variable or addition of an extra constraint.

A constraint $G_j(\boldsymbol{z})$ of formulation *(DC)*,

$$G_j(\boldsymbol{z}) = \sum_{k \in K_j^+} c_{jk} \exp\left(\sum_{i=1}^{N} \alpha_{ijk} z_i\right) - \sum_{k \in K_j^-} c_{jk} \exp\left(\sum_{i=1}^{N} \alpha_{ijk} z_i\right),$$

is guaranteed to be monotonic in z_i if

$$\begin{aligned} & \alpha_{ijk} \geq 0, \ \forall\, k \in K_j^+ \text{ and } \alpha_{ijk} \leq 0, \ \forall\, k \in K_j^- \\ \text{or} \quad & \alpha_{ijk} \leq 0, \ \forall\, k \in K_j^+ \text{ and } \alpha_{ijk} \geq 0, \ \forall\, k \in K_j^-. \end{aligned}$$

However, by appropriately replacing the initial constraint $G_j(\boldsymbol{z})$ with one from the family $\mathcal{G}_j$

$$\begin{aligned} G_j(\boldsymbol{z}, \boldsymbol{d}_j) &= \sum_{k \in K_j^+} c_{jk} \exp\left[\sum_{i=1}^{N} (\alpha_{ijk} + d_{ij})\, z_i\right] \\ &- \sum_{k \in K_j^-} c_{jk} \exp\left[\sum_{i=1}^{N} (\alpha_{ijk} + d_{ij})\, z_i\right], \end{aligned}$$

one can come up with less restrictive conditions for monotonicity. More specifically,

$$G_j(\boldsymbol{z}, \boldsymbol{d}_j) \text{ is monotonically increasing in } z_i \text{ if, } \min_{k \in K_j^+} \alpha_{ijk} \geq \max_{k \in K_j^-} \alpha_{ijk}$$

$$\text{and} \quad d_{ij} \in \left[\max_{k \in K_j^-} \alpha_{ijk}, \min_{k \in K_j^+} \alpha_{ijk}\right]$$

$$G_j(\boldsymbol{z}, \boldsymbol{d}_j) \text{ is monotonically decreasing in } z_i \text{ if, } \max_{k \in K_j^+} \alpha_{ijk} \leq \min_{k \in K_j^-} \alpha_{ijk}$$

$$\text{and} \quad d_{ij} \in \left[\max_{k \in K_j^+} \alpha_{ijk}, \min_{k \in K_j^-} \alpha_{ijk}\right].$$

These monotonicity analysis principles can be utilized to eliminate variables or further constrain formulation *(DC)*.

8.3 GLOBAL OPTIMIZATION ALGORITHM

8.3.1 Description

A global optimization algorithm is proposed for locating the global minimum solution of *(DC)* based on the refinement of converging lower and upper bounds through the solution of convex programming problems *(R)*. The value of the objective function $G_0(\boldsymbol{z})$ at any feasible point $\boldsymbol{z}$ provides an upper bound on the global minimum G_0^*. Lower bounds on G_0^* within some box constraints are derived by solving the relaxed optimization problem *(R)*.

Since *(R)* is convex its single global minimum within some box constraints can be routinely found with a local optimization algorithm and will always underestimate the global minimum of *(DC)* within the same box constraints. Assuming that this global minimum solution of *(R)* is a feasible point for *(DC)*, an upper bound on G_0^* can then be obtained by simply calculating $G_0(\boldsymbol{z})$ at the global minimum point of *(R)*. The gap between the upper bound and the

lower bound on G_0^* will be at most,

$$\begin{aligned}\Delta_{0,max} &= \sum_{k\in K_0^-} c_{0k}\exp\left(Y_{0k}^L\right)\left[1-\frac{\exp\left(\delta_{0k}\right)-1}{\delta_{0k}}\right] \\ &\quad + \sum_{k\in K_0^-} c_{0k}\exp\left(Y_{0k}^L\right)\left[\frac{\exp\left(\delta_{0k}\right)-1}{\delta_{0k}}\log\frac{\exp\left(\delta_{0k}\right)-1}{\delta_{0k}}\right]\end{aligned}$$

where $\delta_{0k} = Y_{0k}^U - Y_{0k}^L$

and $Y_{0k}^L = \sum_{i=1}^N \min\left(\alpha_{i0k}z_i^L, \alpha_{i0k}z_i^U\right),\ Y_{0k}^U = \sum_{i=1}^N \max\left(\alpha_{i0k}z_i^L, \alpha_{i0k}z_i^U\right).$

Note that $\Delta_{0,max}$ goes to zero as all $\delta_{0k},\ k \in K_0^-$ approach zero. Furthermore, because

$$\lim_{\delta_i = z_i^U - z_i^L \to 0^+,\ \forall\, i=1,\dots,N} \delta_{0k} = 0,\ \ \forall\, k \in K_0^-$$

we have

$$\lim_{\delta_i = z_i^U - z_i^L \to 0^+,\ \forall\, i=1,\dots,N} \Delta_{0,max} = 0.$$

This implies that as the current box constraints $[\boldsymbol{z}^L, \boldsymbol{z}^U]$ collapse into a point, the maximum separation $\Delta_{0,max}$ between the original objective function of *(DC)* and its convex relaxation in *(R)* becomes zero. Because $\Delta_{0,max}$ is a continuous function of $\delta_{0k} \geq 0,\ k \in K_0^-$ and therefore of $\delta_i \geq 0,\ i = 1,\dots,N$ this means that for every positive number ϵ_c there exits a positive number δ^* such that if all $\delta_{0k} \leq \delta^*$ we have $\Delta_{0,max} \leq \epsilon_c$. This result reflects the fact that by reducing the bounds $[\boldsymbol{z}^L, \boldsymbol{z}^U]$ the objective function $G_0(\boldsymbol{z})$ of *(DC)* can become arbitrarily close to its convex relaxation $G_0^{conv}(\boldsymbol{z})$ of *(R)*.

By following the same argument the maximum separations $\Delta_{j,max}$ between constraints $G_j(\boldsymbol{z})$ and the corresponding convex relaxations $G_j^{conv}(\boldsymbol{z})$

$$\begin{aligned}\Delta_{j,max} &= \sum_{k\in K_j^-} c_{jk}\exp\left(Y_{jk}^L\right)\left[1-\frac{\exp\left(\delta_{jk}\right)-1}{\delta_{jk}}\right] \\ &\quad + \sum_{k\in K_j^-} c_{jk}\exp\left(Y_{jk}^L\right)\left[\frac{\exp\left(\delta_{jk}\right)-1}{\delta_{jk}}\log\frac{\exp\left(\delta_{jk}\right)-1}{\delta_{jk}}\right]\end{aligned}$$

where $\delta_{jk} = Y_{jk}^U - Y_{jk}^L$

and $Y_{jk}^L = \sum_{i=1}^N \min\left(\alpha_{ijk}z_i^L, \alpha_{ijk}z_i^U\right),\ Y_{jk}^U = \sum_{i=1}^N \max\left(\alpha_{ijk}z_i^L, \alpha_{ijk}z_i^U\right).$

go to zero as all $\delta_{jk}, \forall\, k \in K_j^-$ or alternatively $\delta_i = z_i^U - z_i^L \to 0^+, \forall\, i = 1,\dots,N$ approach zero. Furthermore, $\Delta_{j,max}$ are continuous functions of

δ_{jk}, $k \in K_j^-$ and also of δ_i, $i = 1, \ldots, N$. This implies that for every positive number ϵ_f there exists a positive number δ^* such that by selecting $\delta_{jk} \leq \delta^*$ we have

$$\max_{j \in K_j^-} \Delta_{j,max} \leq \epsilon_f.$$

The interpretation of this result is that by reducing the bounds $[z^L, z^U]$ on z, differences between the feasible region of the original problem *(DC)* and its convex relaxation *(R)* become arbitrarily small. Therefore, any feasible point z^c of problem *(R)* (even the global minimum solution) becomes at least ϵ_f–feasible for problem *(DC)* by sufficiently tightening the bounds on z around this point.

The next step, after establishing an upper and a lower bound on the global minimum, is to tighten them. This is accomplished by restricting the box constraints within the appropriate domain. Tighter box constraints can be realized by partitioning the rectangle into a number of smaller rectangles. The minimum number of variables along which subdivision is required is equal to the number of *linearly independent* $Y'_{jk}s$, $j = 0, \ldots, M$, $k \in K_j^-$. Typically this number is equal to the number of z_i participating in any of the terms $G_j^-(z)$. The following set of variables for which branching is required is defined as :

$$\mathcal{Z} = \left\{ z_{i^*} \ : \ \exists\, \alpha_{i^*jk} \neq 0,\ j = 0, \ldots, M, \text{ and } k \in K_j^- \right\}.$$

In the great majority of cases the number of $z_i \in \mathcal{Z}$ is much smaller than the total number of terms Y_{jk}, $k \in K_j^-$ in formulation *(DC)*, and therefore in this work, subdivision is performed on the set of variables z_i belonging in $\mathcal{Z}$.

One way of partitioning is to successively divide the current rectangle into two subrectangles by halving on the longest side of the initial rectangle (bisection). At each iteration the lower bound of G_0^* is simply the minimum over all the minima of problem *(R)* in every subrectangle enclosing the initial rectangle. Therefore, a straightforward (bound improving) way of tightening the lower bound is to halve at each iteration, only the subrectangle responsible for the infimum of the minima of *(R)* over all subrectangles, according to the rules discussed earlier. This procedure generates a *nondecreasing* sequence for the lower bound of G_0^*. Furthermore, we construct a *nonincreasing* sequence for the upper bound by selecting it to be the infimum over all the previously recorded upper bounds. If the single minimum of *(R)* in any subrectangle is greater than the current upper bound, we can eliminate this subrectangle because the global minimum of $G_0(z)$ cannot be situated inside it. This is also known as the fathoming step.

The next question is how small these subrectangles must become before the upper and lower bounds of $G_0(z)$ are within ϵ_c and also the feasible region of *(DC)* is ϵ_f–close to the feasible region of *(R)*. Because ϵ_c and ϵ_f are very small numbers, the maximum separations $\Delta_{j,max}$, will be proportional to the square of the diagonal δ of the current subrectangle ($\approx \mathcal{O}(\delta^2)$). This means that the required value of δ for convergence is proportional to the square root of ϵ_c.

Therefore, if for example ϵ_c, ϵ_f are set to be 0.0001, it suffices for δ to be of order 0.01.

The basic steps of the proposed global optimization algorithm are summarized in the following subsection.

8.3.2 *Steps of Global Optimization Algorithm for (GGP) Problems*

STEP 1 - *Initialization*

A convergence tolerance ϵ_c and a feasibility tolerance ϵ_f are selected and the iteration counter, $Iter$, is set to one. Appropriate global bounds z_i^{LBD}, z_i^{UBD} on the z_i's are calculated based on the bound improving procedure. Local bounds $z_i^{L,Iter}, z_i^{U,Iter}$ for the first iteration are set to be equal to the global ones. The initial constraints $G_j(\boldsymbol{z})$, $j = 1, \ldots, M$ are replaced by an appropriately selected set of constraints $G_j(\boldsymbol{z}, \boldsymbol{d}_j) \in \mathcal{G}$. In this description of the algorithm, for the sake of simplicity, the parameters $\boldsymbol{d}_j$ are omitted from the constraints. Finally, lower and upper bounds G_0^{LBD}, G_0^{UBD} on the global minimum G_0^* are initialized and an initial current point $z_i^{c,Iter}$ is selected.

STEP 2 - *Feasibility Check and Update of Upper Bound* G_0^{UBD}

If the maximum over all constraints G_j calculated at the current point $z_i^{c,Iter}$ is less than ϵ_f,

$$\max_{j=1,\ldots,M} G_j(\boldsymbol{z}^{c,Iter}) \leq \epsilon_f$$

then the constraint set is ϵ_f–feasible at the current point. If so, the objective function G_0 is calculated at the current point $\boldsymbol{z}^{c,Iter}$ and the upper bound G_0^{UBD} is updated as follows :

$$G_0^{UBD} = \min\left(G_0^{UBD}, G_0(\boldsymbol{z}^{c,Iter})\right).$$

STEP 3 - *Partitioning of Current Rectangle*

The current rectangle $\left[z_i^{L,Iter}, z_i^{U,Iter}\right]$, $i = 1, \ldots, N$ is partitioned into the following two rectangles ($r = 1, 2$):

$$\begin{bmatrix} z_1^{L,Iter} & z_1^{U,Iter} \\ \vdots & \vdots \\ z_{lIter}^{L,Iter} & \dfrac{\left(z_{lIter}^{L,Iter} + z_{lIter}^{U,Iter}\right)}{2} \\ \vdots & \vdots \\ z_N^{L,Iter} & z_N^{U,Iter} \end{bmatrix}, \quad \begin{bmatrix} z_1^{L,Iter} & z_1^{U,Iter} \\ \vdots & \vdots \\ \dfrac{\left(z_{lIter}^{L,Iter} + z_{lIter}^{U,Iter}\right)}{2} & z_{lIter}^{U,Iter} \\ \vdots & \vdots \\ z_N^{L,Iter} & z_N^{U,Iter} \end{bmatrix}$$

where l^{Iter} corresponds to the variable with the longest side in the initial rectangle,

$$l^{Iter} = \arg\max_i \left(z_i^{U,Iter} - z_i^{L,Iter} \right)$$

STEP 4 - *Feasibility Checks of Convex Relaxation (R)*

Update in both subrectangles ($r = 1, 2$) the parameters $Y_{jk}^L, Y_{jk}^U, A_{jk}, B_{jk}$ as follows:

$$Y_{jk}^L = \sum_{i=1}^{N} \min\left(\alpha_{ijk} z_i^L, \alpha_{ijk} z_i^U\right), \; Y_{jk}^U = \sum_{i=1}^{N} \max\left(\alpha_{ijk} z_i^L, \alpha_{ijk} z_i^U\right)$$

$$A_{jk} = \frac{Y_{jk}^U \exp\left(Y_{jk}^L\right) - Y_{jk}^L \exp\left(Y_{jk}^U\right)}{Y_{jk}^U - Y_{jk}^L}, \quad B_{jk} = \frac{\exp\left(Y_{jk}^U\right) - \exp\left(Y_{jk}^L\right)}{Y_{jk}^U - Y_{jk}^L}.$$

If there exists $j = 0, \ldots, M$ such that one of the lower bounds

$$G_j^{conv,L} = \sum_{k \in K_j^+} c_{jk} \exp\left(\sum_{i=1}^{N} Y_{jk}^L\right) - \sum_{k \in K_j^-} c_{jk} \left[A_{jk} + B_{jk}\left(\sum_{i=1}^{N} Y_{jk}^U\right)\right],$$

satisfies

$$G_0^{conv,L} \geq G_0^{UBD} \text{ or } G_j^{conv,L} \geq 0, \text{ for some } j = 1, \ldots, M$$

then the corresponding subrectangle is eliminated (fathoming), go to **STEP 6.** Otherwise, proceed with **STEP 5**.

STEP 5 - *Solution of Convex Problems Inside Subrectangles*

Solve the following convex optimization problem *(R)* in both subrectangles ($r = 1, 2$) by using any nonlinear solver.

$$\begin{aligned} \min_{\boldsymbol{z}} \quad & G_0^{conv}(\boldsymbol{z}) \\ \text{subject to} \quad & G_j^{conv}(\boldsymbol{z}) \leq 0, \; j = 1, \ldots, M \\ & \boldsymbol{z}^L \leq \boldsymbol{z} \leq \boldsymbol{z}^U. \end{aligned}$$

If a solution $G_{0,sol}^{r,Iter}$ is less than the current upper bound, G_0^{UBD} then it is stored along with the value of the variables $\boldsymbol{z}$ at the solution point $\boldsymbol{z}_{i,sol}^{r,Iter}$.

STEP 6 - *Update Iteration Counter Iter and Lower Bound* G_0^{LBD}
The iteration counter is increased by one,

$$Iter \longleftarrow Iter + 1$$

and the lower bound G_0^{LBD} is updated to the minimum solution over the stored ones from previous iterations. Furthermore, the selected solution is erased from the stored set.

$$\begin{aligned} G_0^{LBD} &= G_{0,sol}^{r',Iter'} \\ \text{where} \quad G_{0,sol}^{r',Iter'} &= \min_{r,I} G_{0,sol}^{r,I}, \quad r = 1,2, \quad I = 1,\ldots,Iter-1. \end{aligned}$$

STEP 7 - *Update Current Point $z^{c,Iter}$ and Current Bounds $z^{L,Iter}, z^{U,Iter}$ on z*

The current point is selected to be the solution point of the previously found minimum solution in **STEP 6**,

$$z^{c,Iter} = z_{i,sol}^{r',Iter'}$$

and the current rectangle becomes the subrectangle containing the previously found solution,

$$\left[z_i^{L,Iter}, z_i^{U,Iter}\right] = \begin{bmatrix} z_1^{L,Iter'} & z_1^{U,Iter'} \\ \vdots & \vdots \\ z_{lIter'}^{L,Iter'} & \dfrac{\left(z_{lIter'}^{L,Iter'} + z_{lIter'}^{U,Iter'}\right)}{2} \\ \vdots & \vdots \\ z_N^{L,Iter'} & z_N^{U,Iter'} \end{bmatrix}, \quad \text{if} \quad r' = 1$$

$$\left[z_i^{L,Iter}, z_i^{U,Iter}\right] = \begin{bmatrix} z_1^{L,Iter'} & z_1^{U,Iter'} \\ \vdots & \vdots \\ \dfrac{\left(z_{lIter'}^{L,Iter'} + z_{lIter'}^{U,Iter'}\right)}{2} & z_{lIter'}^{U,Iter'} \\ \vdots & \vdots \\ z_N^{L,Iter'} & z_N^{U,Iter'} \end{bmatrix}, \quad \text{if} \quad r' = 2$$

STEP 8 - *Check for Convergence*

$$\text{If} \quad \left(G_0^{UBD} - G_0^{LBD}\right) > \epsilon_c, \text{ then return to } \textbf{STEP 2}$$

Otherwise, ϵ_c–convergence has been reached and the global minimum, and solution points are:

$$\begin{aligned} G_0^* &\longleftarrow G_0(z^{c,Iter''}) \\ z^* &\longleftarrow z^{c,Iter''} \\ \text{where} \quad Iter'' &= \arg_I \left\{G_0(z^{c,I}) = G_0^{UBD}, \quad I = 1,\ldots,Iter\right\}. \end{aligned}$$

8.4 PROOF OF CONVERGENCE TO THE GLOBAL MINIMUM

In this section, a mathematical proof that the proposed global optimization algorithm converges to the global minimum is given based on the analysis of a standard deterministic global optimization algorithm presented in Horst and Tuy (1990).

In the employed global optimization algorithm, for every partition element $(r, Iter)$ a lower bound, $G_{0,L}^{r,Iter}$, is obtained as the solution of the convex minimization of *(R)* inside the box constraints of the current partition element. Also an upper bound $G_{0,U}^{r,Iter}$ is obtained as the value of G_0 at the single solution of problem *(R)* assuming that it is an ϵ_f–feasible point for *(DC)*. The partition element involving the minimum lower bound $G_{0,L}^{r,Iter}$ is selected for further refinement according to the bisection rule. Partition elements whose lower bound $G_{0,L}^{r,Iter}$ is greater than the current upper bound G_0^{UBD} are fathomed. A sufficient condition for a global optimization algorithm to be convergent to the global minimum, stated in Horst and Tuy (1990), requires that the bounding operation must be *consistent* and the selection operation be *bound improving.*

A bounding operation is called *consistent* if at every step any unfathomed partition can be further refined, and if any infinitely decreasing sequence of successively refined partition elements satisfies,

$$\lim_{Iter\to\infty} \left(G_{0,U}^{Iter} - G_{0,L}^{r,Iter}\right) = 0,$$

where $G_{0,L}^{r,Iter}$ is a lower bound of G_0^* inside the $(r, Iter)$ partition element and $G_{0,L}^{Iter}$ is the best upper bound at iteration $Iter$ not necessarily occurring inside the $(r, Iter)$ partition element. In practice, the requirement that the above limit equals zero for any infinitely decreasing sequence of successively refined partition elements is difficult to verify because $G_{0,U}^{Iter}$ is not necessarily attained inside the partition element $(r, Iter)$. Therefore, in view of the inequality $G_{0,U}^{r,Iter} \geq V_{0,U}^{Iter} \geq G_{0,L}^{r,Iter}$, the following, at least equally strong condition, suffices to be shown:

$$\lim_{Iter\to\infty} \left(G_{0,U}^{r,Iter} - V_{0,L}^{r,Iter}\right) = 0.$$

We have shown that by reducing the size of the partition element $[z^L, z^U]$ where *(R)* is assumed to be feasible, *(DC)* becomes ϵ_f–feasible and the objective function $G_0(z)$ becomes arbitrarily close to its convex relaxation $G_0^{conv}(z)$. This means that,

$$\lim_{\delta_i = z_i^U - z_i^L \to 0^+,\ \forall\, i=1,\ldots,N} G_{0,U}^{r,Iter} - G_{0,L}^{r,Iter} = 0.$$

Therefore, it suffices to show that

$$\lim_{Iter\to\infty} \max_i \delta_i = 0.$$

This is equivalent to requiring that any unfathomed infinitely decreasing sequence of successively refined partition elements is *exhaustive* (Horst and Tuy,

1990). The employed subdivision process is the *bisection* where every partition element $(r, Iter)$ is subdivided into two subrectangles ($r = 1,\ 2$) of equal volume by halving the longest side. By partitioning in this way, each side of every successively refined partition element is halved *exactly* once every K subdivisions. Consequently, after K subdivisions the diagonal of the resulting partition elements is one half the diagonal of the original partition element. Therefore, as the number of successive subdivisions of a partition element goes to infinity, the diagonals of the resulting partition elements go to zero. This implies that the employed subdivision process is *exhaustive.*

Consequently, for any infinitely decreasing sequence of successively refined partition elements

$$\lim_{Iter \to \infty} \left(G_{0,U}^{r,Iter} - G_{0,L}^{r,Iter} \right) = 0$$

meaning that the employed bounding operation is *consistent.*

A *selection operation* is called *bound improving* if at least one partition element where the actual lower bound is attained is selected for further partition after a finite number of refinements. Clearly, the employed selection operation is *bound improving* because the partition element where the actual lower bound is attained is selected for further partitioning in the immediately following iteration.

Hence, we have shown that the bounding operation is *consistent* and that the selection operation is *bound improving*, therefore according to Theorem IV.3. of Horst and Tuy (1990), the employed global optimization algorithm is *convergent* to the global minimum.

8.5 ILLUSTRATIVE EXAMPLE

In this section, an illustrative example is presented and the effect of rescaling on the total number of iterations required for convergence is examined and shown to match the theoretical predictions. The illustrative example, proposed by Avriel *et al.* (1975), involves the minimization of a single variable subject to two nonlinear constraints :

$$\min_{x_1, x_2} \quad x_1$$

$$\text{subject to} \quad \begin{aligned} \frac{1}{4}x_1 + \frac{1}{2}x_2 - \frac{1}{16}x_1^2 - \frac{1}{16}x_2^2 - 1 &\leq 0 \\ \frac{1}{14}x_1^2 + \frac{1}{14}x_2^2 + 1 - \frac{3}{7}x_1 - \frac{3}{7}x_2 &\leq 0 \end{aligned} \qquad (P)$$

$$\begin{aligned} 1 \leq x_1 &\leq 5.5 \\ 1 \leq x_2 &\leq 5.5. \end{aligned}$$

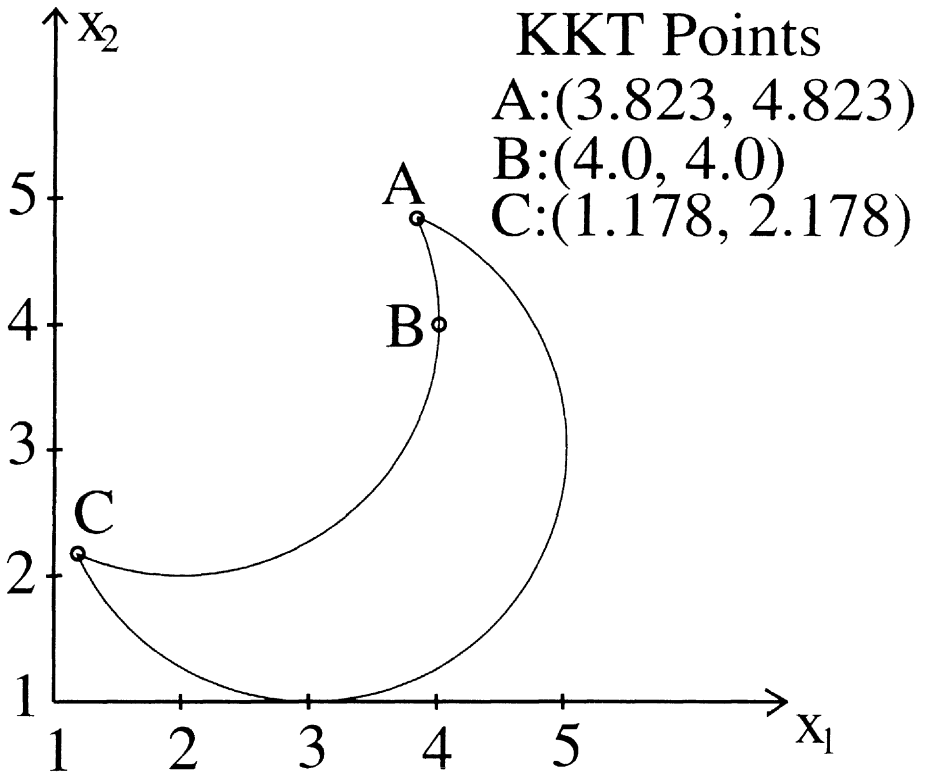


Figure 8.5. Feasible region and KKT points of motivating example

The two nonlinear constraints define a nonconvex crescent–shaped feasible region which is shown in Figure 8.5 and even though the objective function is linear, there still exists three distinct points A,B and C satisfying the Karush–Kuhn–Tucker necessary conditions for optimality. Existing local optimization approaches converge to any of them or erroneously report that the problem is infeasible depending on the location of the initial point.

The following linear variable scaling is then applied on both x_1 and x_2 to show its effect on the total number of iterations for convergence. Note that the effect of this variable rescaling is to maintain tight convex lower bounding of the terms with the form $-\exp(Y)$ by keeping the upper bound on Y as small as possible :

$$x_1 \longleftarrow 1 + \frac{4.5}{U-1}\left(x_1^{new} - 1\right)$$
$$x_2 \longleftarrow 1 + \frac{4.5}{U-1}\left(x_2^{new} - 1\right).$$

This transformation defines a one to one mapping of the original variables x_1, x_2 onto the new variables x_1^{new}, x_2^{new} whose upper bound U is a manipulated parameter :

$$\begin{aligned} x_1^{new} &= 1 + \frac{U-1}{4.5}(x_1 - 1) \\ x_2^{new} &= 1 + \frac{U-1}{4.5}(x_2 - 1) \\ & 1 \le x_1^{new} \le U \\ & 1 \le x_2^{new} \le U. \end{aligned}$$

After substituting the expressions for x_1 and x_2 we obtain the following transformed problem, which is parametric in U:

$$\min_{x_1^{new}, x_2^{new}} \quad 1 + \frac{4.5}{U-1}(x_1^{new} - 1)$$

$$\text{subject to} \quad \begin{aligned} & \frac{63+18U}{32(U-1)^2}x_1^{new} + \frac{27+54U}{32(U-1)^2}x_2^{new} - \frac{81}{64(U-1)^2}(x_1^{new})^2 \\ & -\frac{81}{64(U-1)^2}(x_2^{new})^2 - \frac{21+48U+12U^2}{32(U-1)^2} \le 0 \qquad (P_U) \\ & \frac{81}{56(U-1)^2}(x_1^{new})^2 + \frac{81}{56(U-1)^2}(x_2^{new})^2 + \frac{17+56U+8U^2}{28(U-1)^2} \\ & -\frac{45+36U}{28(U-1)^2}x_1^{new} - \frac{45+36U}{28(U-1)^2}x_2^{new} \le 0 \end{aligned}$$

$$\begin{aligned} 1 &\le x_1^{new} \le U \\ 1 &\le x_2^{new} \le U. \end{aligned}$$

For any value of the new upper bound U ($U > 1$), problem (P_U) has the same solution as *(P)*. Furthermore, the performed variable scaling does not affect the scaling of the constraints. Note that even though the scaling of the new variables x_1^{new} and x_2^{new} depends on U, the values of the objective function and the constraints remain unchanged in (P_U). Next, the exponential variable transformation,

$$x_i^{new} = \exp z_i, \quad i = 1, 2$$

is applied. This transformation induces a difference of two convex functions structure in both objective function and constraints. The reformulated problem (D_U) in the space of the two variables z_1, z_2, which is parametric in U and completely equivalent with P_U, can be written as follows:

$$\min_{z_1,z_2} \quad \frac{4.5}{U-1}\exp z_1 + \left(1 - \frac{4.5}{U-1}\right)$$

$$\begin{aligned}
\text{subject to} \quad & \frac{63+18U}{32(U-1)^2}\exp z_1 + \frac{27+54U}{32(U-1)^2}\exp z_2 - \frac{81}{64(U-1)^2}\exp 2z_1 \\
& -\frac{81}{64(U-1)^2}\exp 2z_2 - \frac{21+48U+12U^2}{32(U-1)^2} \leq 0 \qquad (D_U) \\
& \frac{81}{56(U-1)^2}\exp 2z_1 + \frac{81}{56(U-1)^2}\exp 2z_2 + \frac{17+56U+8U^2}{28(U-1)^2} \\
& -\frac{45+36U}{28(U-1)^2}\exp z_1 - \frac{45+36U}{28(U-1)^2}\exp z_2 \leq 0
\end{aligned}$$

$$\begin{aligned}
0 &\leq z_1^{new} \leq \log U \\
0 &\leq z_2^{new} \leq \log U.
\end{aligned}$$

To further analyze the effect of the exponential transformation consider the first inequality constraint of (D_U) with $U = 5.5$.

$$(0.25\exp z_1 + 0.5\exp z_2) \;-\; (0.0625\exp 2z_1 + 0.0625\exp 2z_2 + 1) \;\leq\; 0.$$

This can be viewed as the difference of the two convex functions $(0.25\exp z_1 + 0.5\exp z_2)$ and $(0.0625\exp 2z_1 + 0.0625\exp 2z_2 + 1)$.

A convex relaxation (R_U) of (D_U) is then derived by lower bounding the concave terms $-\exp z_i, -\exp 2z_i$ of problem (D_U) with linear segments connecting the two end points $z_i^L = 0, z_i^U = \log U$:

$$-\exp z_i \longleftarrow \frac{1-\exp U}{\log U} z_i - 1, \quad i = 1,2$$

$$-\exp 2z_i \longleftarrow \frac{1-\exp 2U}{\log U} z_i - 1, \quad i = 1,2.$$

Note that even though the solution of (D_U) does not depend on the selection of U, the solution of the convex relaxation (R_U) of (D_U) is indeed a function of U. This implies that different values for U will yield a different number of required iterations for convergence. After selecting the convergence and the feasibility tolerances to be equal to $\epsilon_c = \epsilon_f = 10^{-4}$, the proposed global optimization procedure is applied and convergence to the global minimum point $(x_1^* = 1.178, x_2^* = 2.178)$ is achieved from all starting points for different values of U. The number of iterations as a function of U are summarized in Table 8.2. It appears that the total number of iterations is a monotonically increasing function of U. This implies that the tighter the rescaling is the better the underestimation will be which is consistent with the theoretical results. Note

U	Iterations
1.1	17
1.5	18
2	19
5.5	20
10	21
100	27
1000	58

Table 8.2. Number of iterations as a function of U

that as U approaches one, numerical stability problems offset the benefits of marginally better underestimators.

Summary

In this chapter, we discussed a deterministic branch and bound type global optimization algorithm for solving generalized geometric programming problems *(GGP)* studied by Maranas and Floudas (1997). An exponential variable transformation was employed to the initial nonconvex problem *(GGP)* to reduce it into a *(DC)* programming problem. A convex relaxation *(R)* is then obtained based on the linear lower bounding of the concave parts of the objective function and constraints. The proposed algorithm was shown to attain finite ϵ–convergence to the global minimum through the successive refinement of a convex relaxation of the feasible region and/or of the objective function and the subsequent solution of a series of nonlinear convex optimization problems. A number of procedures aimed at improving the efficiency of the proposed approach are also discussed, including the elimination of variables through monotonicity analysis, variable rescaling, variable bound improving, and inequality transformation.

9 GENERALIZED GEOMETRIC PROGRAMMING : COMPUTATIONAL STUDIES

This chapter presents extensive computational studies of the global optimization approach for generalized geometric programming *(GGP)* problems presented in Chapter 8. A number of engineering design problems are addressed for which convergence to an ϵ-global minimum is guaranteed. A framework for checking the stability of given control strategies is presented which requires the solution of *(GGP)* problems, and a number of stability analysis examples are considered. The deterministic global optimization algorithm introduced in Chapter 8 has been implemented with the modeling language GAMS (Brooke *et al.*, 1992) and computational times are reported for all examples on a HP–730 workstation.

9.1 CHEMICAL ENGINEERING DESIGN EXAMPLES

In this section, chemical engineering example problems that are modeled as generalized geometric programming problems are presented and the results of the application of the global optimization approach presented in Chapter 8 are discussed. The following examples correspond to an alkylation process, a CSTR sequence, a heat exchanger and an optimal reactor design. Additional computational studies of generalized geometric programming *(GGP)* problems can be found in Maranas and Floudas (1997) and in the recent collection of test problems in local and global optimization by Floudas *et al.* (1999).

C. A. Floudas, *Deterministic Global Optimization*
© Springer Science+Business Media Dordrecht 2000

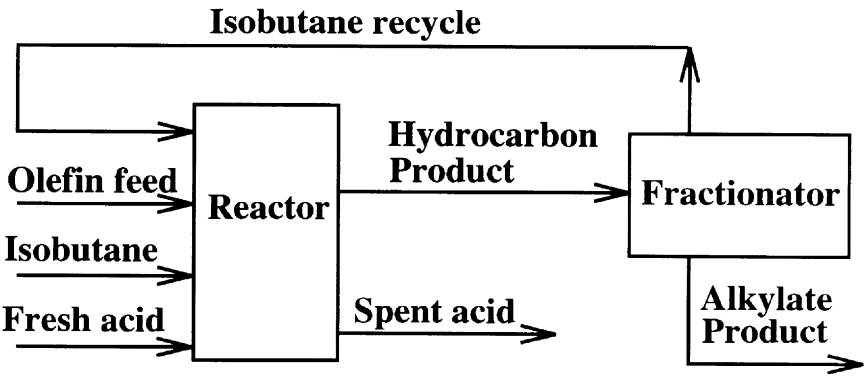


Figure 9.1. Alkylation unit diagram

9.1.1 Alkylation Process Design

This example involves the design of an alkylation unit as modeled by Dembo (1976). The objective is to improve the octane number of some olefin feed by reacting it with isobutane in the presence of acid. The product of the reaction is distilled and the unreacted isobutane is recycled back to the reactor (see Figure 9.1).

All equality constraints are eliminated and the problem has been formulated as the following signomial optimization problem:

$$\min \quad c_1x_1 + c_2x_1x_6 + c_3x_3 + c_4x_2 + c_5 - c_6x_3x_5$$

$$
\begin{aligned}
\text{subject to} \qquad c_7x_6^2 + c_8x_1^{-1}x_3 - c_9x_6 &\leq 1 \\
c_{10}x_1x_3^{-1} + c_{11}x_1x_3^{-1}x_6 - c_{12}x_1x_3^{-1}x_6^2 &\leq 1 \\
c_{13}x_6^2 + c_{14}x_5 - c_{15}x_4 - c_{16}x_6 &\leq 1 \\
c_{17}x_5^{-1} + c_{18}x_5^{-1}x_6 + c_{19}x_4x_5^{-1} - c_{20}x_5^{-1}x_6^2 &\leq 1 \\
c_{21}x_7 + c_{22}x_2x_3^{-1}x_4^{-1} - c_{23}x_2x_3^{-1} &\leq 1 \\
c_{24}x_7^{-1} + c_{25}x_2x_3^{-1}x_7^{-1} - c_{26}x_2x_3^{-1}x_4^{-1}x_7^{-1} &\leq 1 \\
c_{27}x_5^{-1} + c_{28}x_5^{-1}x_7 &\leq 1 \\
c_{29}x_5 - c_{30}x_7 &\leq 1 \\
c_{31}x_3 - c_{32}x_1 &\leq 1 \\
c_{33}x_1x_3^{-1} + c_{34}x_3^{-1} &\leq 1 \\
c_{35}x_2x_3^{-1}x_4^{-1} - c_{36}x_2x_3^{-1} &\leq 1 \\
c_{37}x_4 + c_{38}x_2^{-1}x_3x_4 &\leq 1 \\
c_{39}x_1x_6 + c_{40}x_1 - c_{41}x_3 &\leq 1 \\
c_{42}x_1^{-1}x_3 + c_{43}x_1^{-1} - c_{44}x_6 &\leq 1
\end{aligned}
$$

$$
\begin{array}{rcccr}
1500 & \leq & x_1 & \leq & 2000 \\
1 & \leq & x_2 & \leq & 120 \\
3000 & \leq & x_3 & \leq & 3500 \\
85 & \leq & x_4 & \leq & 93 \\
90 & \leq & x_5 & \leq & 95 \\
3 & \leq & x_6 & \leq & 12 \\
145 & \leq & x_7 & \leq & 162
\end{array}
$$

where c_i, $i = 1, \ldots, 44$ are positive parameters given in Table 9.1. This problem involves seven nonlinear variables subject to 12 nonlinear and two linear inequality constraints. Note that there exists a large number of bilinear, trilinear, and even fractional terms in the objective function and constraints. First, the bound improving procedure yields the following reduced variable bounds:

$$
\begin{array}{rcccr}
1660.00 & \leq & x_1 & \leq & 2000.00 \\
23.54 & \leq & x_2 & \leq & 120.00 \\
3000.00 & \leq & x_3 & \leq & 3278.68 \\
85.00 & \leq & x_4 & \leq & 93.00 \\
92.18 & \leq & x_5 & \leq & 95.00 \\
3.78 & \leq & x_6 & \leq & 10.68 \\
145.00 & \leq & x_7 & \leq & 153.53.
\end{array}
$$

i	c_i	i	c_i	i	c_i
1	1.715	16	0.19120592 E-1	31	0.00061000
2	0.035	17	0.56850750 E+2	32	0.0005
3	4.0565	18	1.08702000	33	0.81967200
4	10.000	19	0.32175000	34	0.81967200
5	3000.0	20	0.03762000	35	24500.0
6	0.063	21	0.00619800	36	250.0
7	0.59553571 E-2	22	0.24623121 E+4	37	0.10204082 E-1
8	0.88392857	23	0.25125634 E+2	38	0.12244898 E-4
9	0.11756250	24	0.16118996 E+3	39	0.00006250
10	1.10880000	25	5000.0	40	0.00006250
11	0.13035330	26	0.48951000 E+6	41	0.00007625
12	0.00660330	27	0.44333333 E+2	42	1.22
13	0.66173269 E-3	28	0.33000000	43	1.0
14	0.17239878 E-1	29	0.02255600	44	1.0
15	0.56595559 E-2	30	0.00759500		

Table 9.1. Coefficients for alkylation example

Convergence to the global minimum point,

$$\begin{aligned} x_1^* &= 1698.18 \\ x_2^* &= 53.66 \\ x_3^* &= 3031.30 \\ x_4^* &= 90.11 \\ x_5^* &= 95 \\ x_6^* &= 10.50 \\ x_7^* &= 153.53 \end{aligned}$$

having an objective function value of $f^* = 1227.23$ is reached in about 200 iterations and 30 seconds of CPU time.

9.1.2 CSTR Sequence Design with Capital Cost Constraints

This example (see Figure 9.2) was first posed by Manousiouthakis and Sourlas (1992) and involves the design of a sequence of two CSTR reactors where the consecutive reaction $A \rightarrow B \rightarrow C$ takes place. The concentration C_{b2} of B in the outlet stream of the second reactor needs to be maximized subject to an

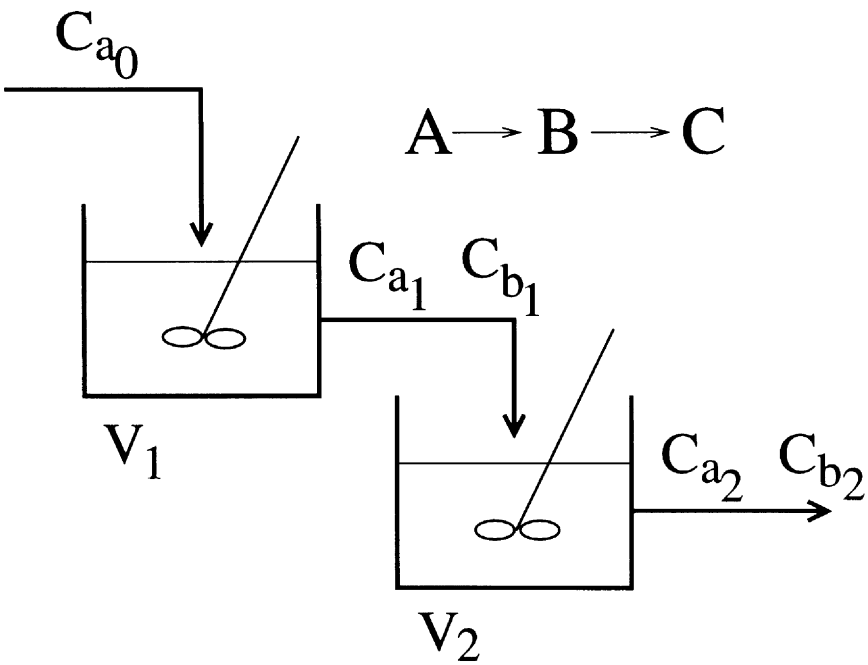


Figure 9.2. Two interlinked CSTR's

upper bound on the investment cost. First order kinetics are assumed for both reactions and the reaction constants for the first (a) and second (b) reaction in reactors (1) and (2) are

$$
\begin{aligned}
k_{a1} &= 9.6540\ 10^{-2}\ s^{-1}, \\
k_{b1} &= 3.5272\ 10^{-2}\ s^{-1}, \\
k_{a2} &= 9.7515\ 10^{-2}\ s^{-1}, \\
k_{b2} &= 3.9191\ 10^{-2}\ s^{-1}.
\end{aligned}
$$

The inlet concentration of A is $c_{a0} = 1.0 mol/l$, and zero for both B and C. If V_1, V_2 are the residence times for the two reactors and c_{a1}, c_{b1}, c_{a2}, and c_{b2} the corresponding exit concentrations, then the reactor design problem can be formulated as the following nonconvex optimization problem:

$$\min \quad -c_{b2}$$

$$\begin{aligned}
\text{subject to} \qquad (c_{a1} - c_{a0}) + k_{a1}c_{a1}V_1 &= 0 \\
(c_{a2} - c_{a1}) + k_{a2}c_{a2}V_2 &= 0 \\
(c_{b1} + c_{a1} - c_{a0}) + k_{b1}c_{b1}V_1 &= 0 \\
(c_{b2} - c_{b1} + c_{a2} - c_{a1}) + k_{b2}c_{b2}V_2 &= 0 \\
V_1^{0.5} + V_2^{0.5} &\leq 4
\end{aligned}$$

$$0 \leq c_{a1}, c_{b1}, c_{a2}, c_{b2} \leq 1$$

$$1 \leq V_1, V_2 \leq 100.$$

Assuming that the capital cost of a reactor is proportional to the square root of its residence time, constraint $V_1^{0.5} + V_2^{0.5} \leq 4$ provides an upper bound on the total investment costs. Also, the last two constraints define some initial bounds on the variables. This problem has been reported to have two local minima (Manousiouthakis and Sourlas, 1992).

First, by applying the bound improving procedure discussed in Chapter 8 the initial bounds are refined into the following:

$$\begin{array}{rcl}
0.4090 \leq c_{a1} \leq 0.9119 \\
0.2128 \leq c_{a2} \leq 0.8309 \\
0.0845 \leq c_{b1} \leq 0.4501 \\
0.0571 \leq c_{b2} \leq 0.7099 \\
1.0000 \leq V_1 \leq 9.0000 \\
1.0000 \leq V_2 \leq 9.0000.
\end{array}$$

Application of the global optimization approach presented in Chapter 8 yields the following global optimum solution

$$\begin{aligned}
c_{a1}^* &= 0.6587 \\
c_{a2}^* &= 0.5161 \\
c_{b1}^* &= 0.2869 \\
c_{b2}^* &= 0.3866 \\
V_1^* &= 5.3666 \\
V_2^* &= 2.8338
\end{aligned}$$

in about 400 iterations and a CPU time of 20 seconds with convergence and feasibility tolerances of 10^{-4}.

In the following, the bounds of the reactor volumes are expanded to:

$$10^{-6} \leq V_1, V_2 \leq 16$$

and all concentration variables c_{a1}, c_{a2}, c_{b1} and c_{b2} are eliminated from the initial formulation for the reactor design problem. The resulting compact formulation involves only two variables V_1, V_2 and takes the form:

$$\min \quad -c_{b2} = -c_{a0} \frac{k_{a2}V_2\,(1+k_{b1}V_1) + k_{a1}V_1\,(1+k_{a2}V_2)}{(1+k_{a1}V_1)\,(1+k_{b1}V_1)\,(1+k_{a2}V_2)\,(1+k_{b2}V_2)}$$

$$\begin{aligned} \text{subject to} \quad V_1^{0.5} + V_2^{0.5} &\leq 4 \\ 10^{-6} \leq V_1, V_2 &\leq 16. \end{aligned}$$

Note that unlike methods based on the bilinearization of the original problem (Manousiouthakis and Sourlas, 1992; Ryoo and Sahinidis, 1995), the global optimization approach presented in Chapter 8 can readily handle ratios of polynomial expressions and thus take advantage of the reduction of the total number of variables from six to only two. After transforming the fractional objective into a signomial form it takes only about 282 iterations and 17 seconds of CPU to locate the global optimum solution ($V_1^* = 15.992 \approx 16, V_2^* = 10^{-6} \approx 0$). This corresponds to over an order of magnitude reduction in the computational requirements reported in Manousiouthakis and Sourlas (1992), (7950 iterations and 7950 seconds on an Apollo DN10000). This is attributed to the fact that not only are additional variables not introduced but also elimination of existing variables is possible. Consequently, the obtained lower bounds are much tighter and the branch and bound procedure is performed on a smaller set of variables.

Ryoo and Sahinidis (1995) solved globally the same problem by convex lower bounding the bilinear terms for a different set of reaction constants which are :

$$\begin{aligned} k_{a1} &= 9.755988\,10^{-2}\ s^{-1} \\ k_{b1} &= 3.919080\,10^{-2}\ s^{-1} \\ k_{a2} &= 9.658428\,10^{-2}\ s^{-1} \\ k_{b2} &= 3.527172\,10^{-2}\ s^{-1}. \end{aligned}$$

A number of variations on the main algorithm were considered and the best one yielded a CPU time of 23 seconds on a SPARC 2.
The global optimization approach presented in Chapter 8, converged to the global optimum ($V_1^* = 3.037, V_2^* = 5.096$) after 299 iterations and about 20 seconds of CPU time.

9.1.3 Heat Exchanger Design

This standard heat exchanger design problem was first formulated as a generalized geometric programming problem by Avriel and Williams (1971). It can be formulated as follows:

$$\min \quad x_1 + x_2 + x_3$$

$$
\begin{aligned}
\text{subject to} \quad 833.33252x_1^{-1}x_4x_6^{-1} + 100.0x_6^{-1} - 83333.333x_1^{-1}x_6^{-1} &\leq 1 \\
1250.0x_2^{-1}x_5x_7^{-1} + 1.0x_4x_7^{-1} - 1250.0x_2^{-1}x_4x_7^{-1} &\leq 1 \\
1250000.0x_3^{-1}x_8^{-1} + 1.0x_5x_8^{-1} - 2500.0x_3^{-1}x_5x_8^{-1} &\leq 1 \\
0.0025x_4 + 0.0025x_6 &\leq 1 \\
0.0025x_5 + 0.0025x_6 - 0.0025x_7 &\leq 1 \\
0.01x_8 - 0.01x_5 &\leq 1
\end{aligned}
$$

$$
\begin{array}{rcccl}
500 & \leq & x_1 & \leq & 1000 \\
1000 & \leq & x_2 & \leq & 5000 \\
5000 & \leq & x_3 & \leq & 10000 \\
500 & \leq & x_4 & \leq & 1000 \\
500 & \leq & x_5 & \leq & 1000 \\
500 & \leq & x_6 & \leq & 1000 \\
500 & \leq & x_7 & \leq & 1000 \\
500 & \leq & x_8 & \leq & 1000.
\end{array}
$$

The bound improving procedure considerably reduces the initial bounds to:

$$
\begin{array}{rcccl}
500 & \leq & x_1 & \leq & 1000.0 \\
1000 & \leq & x_2 & \leq & 1550.00 \\
5000 & \leq & x_3 & \leq & 5550.00 \\
126.02 & \leq & x_4 & \leq & 231.69 \\
271.35 & \leq & x_5 & \leq & 328.04 \\
107.31 & \leq & x_6 & \leq & 270.54 \\
238.78 & \leq & x_7 & \leq & 343.97 \\
371.53 & \leq & x_8 & \leq & 426.92.
\end{array}
$$

The global optimization approach presented in Chapter 8 generates the global minimum point

$$
\begin{array}{rcr}
x_1^* & = & 579.31 \\
x_2^* & = & 1359.97 \\
x_3^* & = & 5109.97 \\
x_4^* & = & 182.01 \\
x_5^* & = & 295.60 \\
x_6^* & = & 217.98 \\
x_7^* & = & 286.42 \\
x_8^* & = & 395.60
\end{array}
$$

with an objective function value of $f^* = 7049.24$ in about 1600 iterations and a CPU time of 100 seconds.

9.1.4 Optimal Reactor Design

This problem was proposed by Dembo (1976) and it addresses the optimal design of a reactor. It involves a signomial objective function and posynomial constraints and takes the form :

$$\begin{array}{lrcl} \min & 0.4x_1^{0.67}x_7^{-0.67} + 0.4x_2^{0.67}x_8^{-0.67} + 10.0 - x_1 - x_2 & & \\ \text{subject to} & 0.0588x_5x_7 + 0.1x_1 & \leq & 1 \\ & 0.0588x_6x_8 + 0.1x_1 + 0.1x_2 & \leq & 1 \\ & 4x_3x_5^{-1} + 2x_3^{-0.71}x_5^{-1} + 0.0588x_3^{-1.3}x_7 & \leq & 1 \\ & 4x_4x_6^{-1} + 2x_4^{-0.71}x_6^{-1} + 0.0588x_4^{-1.3}x_8 & \leq & 1 \end{array}$$

$$0.1 \leq x_i \leq 10, \quad i = 1, \ldots, 8.$$

The bound improving procedure yields:

$$\begin{array}{rcccr} 0.1 & \leq & x_1 & \leq & 9.5868 \\ 0.1 & \leq & x_2 & \leq & 9.5868 \\ 0.1274 & \leq & x_3 & \leq & 2.2100 \\ 0.1274 & \leq & x_4 & \leq & 2.2100 \\ 5.3252 & \leq & x_5 & \leq & 10.0 \\ 5.3252 & \leq & x_6 & \leq & 10.0 \\ 0.1 & \leq & x_7 & \leq & 2.4529 \\ 0.1 & \leq & x_8 & \leq & 2.4327. \end{array}$$

Convergence to the global minimum point

$$\begin{array}{rcl} x_1^* & = & 6.4747 \\ x_2^* & = & 2.2340 \\ x_3^* & = & 0.6671 \\ x_4^* & = & 0.5957 \\ x_5^* & = & 5.9310 \\ x_6^* & = & 5.5271 \\ x_7^* & = & 1.0108 \\ x_8^* & = & 0.4004 \end{array}$$

with an objective function value $f^* = 3.9511$ was achieved in 71 iterations and a CPU time of 6.8 seconds.

9.2 ROBUST STABILITY ANALYSIS

Robust stability analysis of linear systems involves the identification of the largest possible region in the uncertain model parameter space for which the controller manages to attenuate any disturbances in the system. The stability of a feedback structure is determined by the roots of the closed loop characteristic equation:

$$\det\left(I + P(s, \boldsymbol{q})C(s, \boldsymbol{q})\right) = 0$$

where $\boldsymbol{q}$ is the vector of the uncertain model parameters, and $P(s) and C(s)$ are the transfer functions of the plant and controller, respectively. After expanding the determinant we have the characteristic polynomial:

$$P(s, \boldsymbol{q}) = a_n(\boldsymbol{q})s^n + a_{n-1}(\boldsymbol{q})s^{n-1} + \cdots + a_1(\boldsymbol{q})s + a_0(\boldsymbol{q}) = 0$$

where the coefficients $a_i(\boldsymbol{q})$, $i = 0, \ldots, n$ are typically multivariable polynomial functions. The "zero exclusion condition" (ZEC) implies that a system with characteristic equation $P(s, \boldsymbol{q}) = 0$ is stable only if it does not have any roots on the imaginary axis for any realization of the $\boldsymbol{q}$'s in the uncertain model parameter space $\mathcal{Q}$, that is :

$$0 \notin P(j\omega, \boldsymbol{q}), \ \forall \boldsymbol{q} \in \mathcal{Q}, \text{ and } \forall \omega \in [0, \infty].$$

A stability margin k_m can then be defined as follows:

$$k_m(j\omega) = \inf\{k : P(j\omega, \boldsymbol{q}(k)) = 0, \forall \boldsymbol{q} \in \mathcal{Q}\}.$$

Robust stability for this model is then guaranteed if and only if:

$$k_m \geq 1.$$

Geometrically, k_m expands the initial uncertain parameter region $\mathcal{Q}$ as much as possible without loosing stability (See Figure 9.3). Note that real parameter uncertainty is typically expressed as positive and negative deviations of the real parameters from some nominal values. Checking the stability of a particular system with characteristic equation $P(j\omega, \boldsymbol{q})$ involves the solution of the following nonconvex optimization problem (Psarris and Floudas, 1995) :

$$\min_{q_i, k \geq 0, \omega \geq 0} \quad k \qquad (S)$$

$$\begin{aligned} \text{subject to} \quad Re\,[P(j\omega, \boldsymbol{q})] &= 0 \\ Im\,[P(j\omega, \boldsymbol{q})] &= 0 \\ q_i^N - \Delta q_i^- \, k \leq q_i &\leq q_i^N + \Delta q_i^+ \, k, \quad i = 1, \ldots, n \end{aligned}$$

where $\boldsymbol{q}^N$ is a stable nominal point for the uncertain parameters and $\Delta\boldsymbol{q}^+, \Delta\boldsymbol{q}^-$ are the estimated bounds. Note that it is important to be able to always locate the global minimum of *(S)*, otherwise the stability margin might be overestimated. This overestimation can sometimes lead to the erroneous conclusion that a system is stable when it is not. Because for most problems without time delays $a_i(\boldsymbol{q})$, $i = 0, \ldots, n$ are multivariable polynomial functions, formulation *(S)* corresponds to a generalized geometric programming problem. In the following, a number of stability analysis examples are considered.

9.2.1 Example 1

This example was first proposed by Gaston and Safonov (1988). The plant has three uncertain parameters and the characteristic polynomial equation is:

$$\begin{aligned} P(s, q_1, q_2, q_3) &= s^4 + (10 + q_2 + q_3)\, s^3 + (q_2 q_3 + 10 q_2 + 10 q_3)\, s^2 \\ &+ (1 - q_2 q_3 + q_1)\, s + 2 q_1. \end{aligned}$$

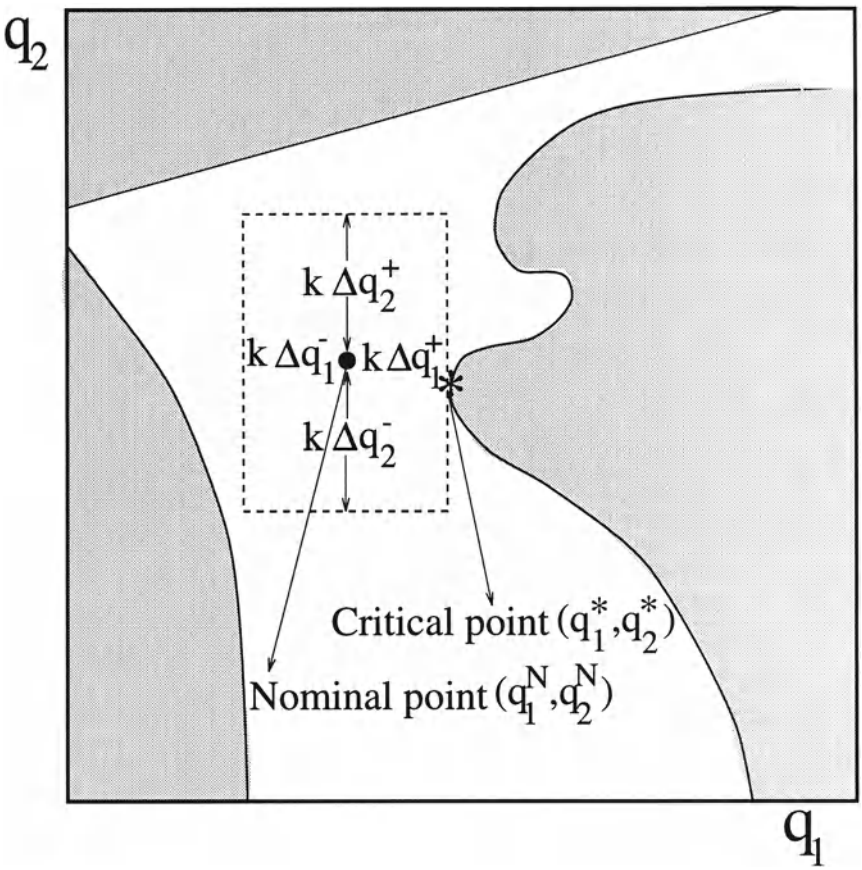


Figure 9.3. Stability margin

The nominal values of the parameters of the system are

$$q_1^N = 800,\ q_2^N = 4,\ q_3^N = 6$$

and the bounded perturbations

$$\Delta q_1^+ = \Delta q_1^- = 800,\ \Delta q_2^+ = \Delta q_2^- = 2,\ \Delta q_3^+ = \Delta q_3^- = 3.$$

After eliminating ω, the zero exclusion formulation becomes:

$$\min \; k$$

$$\begin{aligned}
\text{subject to} \quad & 10q2^2q_3^3 + 10q_2^3q_3^2 + 200q_2^2q_3^2 + 100q_2^3q_3 \\
& +100q_2q_3^3 + q_1q_2q_3^2 + q_1q_2^2q_3 + 1000q_2q_3^2 \\
& +8q_1q_3^2 + 1000q_2^2q_3 + 8q_1q_2^2 + 6q_1q_2q_3 \\
& +60q_1q_3 + 60q_1q_2 - q_1^2 - 200q_1 \;\leq\; 0
\end{aligned}$$

$$\begin{array}{rcccl}
800 - 800k & \leq & q_1 & \leq & 800 + 800k \\
4 - 2k & \leq & q_2 & \leq & 4 + 2k \\
6 - 3k & \leq & q_3 & \leq & 6 + 3k.
\end{array}$$

The stability margin is found to be equal to $k_m = 0.3417$, which implies that the system is unstable. Furthermore, the first instability occurs at:

$$\begin{array}{rcr}
q_1^* & = & 1073.4 \\
q_2^* & = & 3.318 \\
q_3^* & = & 4.975.
\end{array}$$

Computational requirements were 15 iterations and 0.5 seconds of CPU time.

9.2.2 Example 2

This example examines the l_∞ stability margin for the closed–loop system described in Sideris and Sànchez (1988). The characteristic polynomial is:

$$P(s, q_1, q_2, q_3) \;=\; s^4 + \left(q_1^3q_2\right)s^3 + \left(q_1^2q_2^2q_3\right)s^2 + \left(q_1q_2^3q_3^2\right)s + q_3^3.$$

The nominal parameter values are

$$q_1^N = 1.4,\; q_2^N = 1.5,\; q_3^N = 0.8$$

and the bounded perturbations

$$\Delta q_1^+ = \Delta q_1^- = 0.25,\; \Delta q_2^+ = \Delta q_2^- = 0.20,\; \Delta q_3^+ = \Delta q_3^- = 0.20.$$

Eliminating ω the zero exclusion formulation becomes:

$$\min \; k$$

$$\text{subject to} \quad q_1^4q_2^4 - q_1^4 - q_2^4q_3 \;=\; 0$$

$$\begin{array}{rcccl}
1.4 - 0.25k & \leq & q_1 & \leq & 1.4 + 0.25k \\
1.5 - 0.20k & \leq & q_2 & \leq & 1.5 + 0.20k \\
0.8 - 0.20k & \leq & q_3 & \leq & 0.8 + 0.20k.
\end{array}$$

The stability margin is found to be equal to $k_m = 1.089$, implying that the system is stable. Furthermore, the unstable point closest to the nominal point is:

$$\begin{aligned} q_1^* &= 1.1275 \\ q_2^* &= 1.2820 \\ q_3^* &= 1.0179. \end{aligned}$$

Computational requirements were 26 iterations and 1.4 seconds of CPU time.

9.2.3 Example 3

This example involves affine coefficient functions and it was first proposed by Ackermann *et al.* (1991). The closed–loop characteristic polynomial is:

$$\begin{aligned} P(s, q_1, q_2, q_3) = \quad & 16s^4 + (16q_1 - 24)\, s^3 + (-9.625q_1 - 16q_2 + 78)\, s^2 \\ + \quad & (19q_1 + 8q_2 + q_3 - 44)\, s + (q_3 - q_2 - 4q_1 + 12)\,. \end{aligned}$$

The nominal parameter values are

$$q_1^N = 2.25,\ q_2^N = 1.5,\ q_3^N = 1.5$$

and the bounded perturbations

$$\Delta q_1^+ = \Delta q_1^- = 0.25,\ \Delta q_2^+ = \Delta q_2^- = 0.50,\ \Delta q_3^+ = \Delta q_3^- = 1.50.$$

In this example the frequency, ω, is not eliminated and therefore the formulation is:

$$\min \ k$$

$$\begin{aligned} \text{subject to} \quad q_3 + 9.625q_1\omega + 16q_2\omega + 16\omega^2 + 12 - 4q_1 - q_2 - 78\omega &= 0 \\ 16q_1\omega + 44 - 19q_1 - 8q_2 - q_3 - 24\omega &= 0 \end{aligned}$$

$$\begin{aligned} 2.25 - 0.25k &\le q_1 \le 2.25 + 0.25k \\ 1.5 - 0.50k &\le q_2 \le 1.5 + 0.50k \\ 1.5 - 1.50k &\le q_3 \le 1.5 + 1.50k. \end{aligned}$$

The stability margin is found to be equal to $k_m = 0.8175$ which means that the system is unstable. Note that application of the local solver MINOS 5.4 consistently overestimated the stability margin to be $k_m = 0.8765$. The critical parameter values for instability are:

$$\begin{aligned} q_1^* &= 2.4544 \\ q_2^* &= 1.9088 \\ q_3^* &= 2.7263 \\ \omega^* &= 1.3510. \end{aligned}$$

Computational requirements were 51 iterations and 5 seconds of CPU time.

9.2.4 Example 4

This example studies the stability of the mechanical system introduced by Vicino *et al.* (1990). The corresponding closed–loop characteristic polynomial is:

$$P(s,\boldsymbol{q}) = a_4(\boldsymbol{q})s^4 + a_3(\boldsymbol{q})s^3 + a_2(\boldsymbol{q})s^2 + a_1(\boldsymbol{q})s^1 + a_0(\boldsymbol{q})$$

$$\begin{aligned}
\text{where } a_4(\boldsymbol{q}) &= q_3^2 q_2\,(4q_2 + 7q_1) \\
a_3(\boldsymbol{q}) &= 7q_4q_3^2q_2 - 64.918q_3^2q_2 + 380.067q_3q_2 + 3q_5q_2 + 3q_5q_1 \\
a_2(\boldsymbol{q}) &= 3\left(-9.81q_3q_2^2 - 9.81q_3q_1q_2 - 4.312q_3^2q_2 + 264.896q_3q_2\right) \\
&+ 3\left(q_4q_5 - 9.274q_5\right) \\
a_1(\boldsymbol{q}) &= \frac{1}{5}\left(-147.15q_4q_3q_2 + 1364.67q_3q_2 - 27.72q_5\right) \\
a_0(\boldsymbol{q}) &= 54.387q_3q_2.
\end{aligned}$$

The nominal parameter values are

$$q_1^N = 10.0,\ q_2^N = 1.0,\ q_3^N = 1.0,\ q_4^N = 0.2,\ q_5^N = 0.05$$

and the parameter perturbations

$$\begin{aligned}
\Delta q_1^+ = \Delta q_1^- &= 1.000, \\
\Delta q_2^+ = \Delta q_2^- &= 0.100, \\
\Delta q_3^+ = \Delta q_3^- &= 0.100, \\
\Delta q_4^+ = \Delta q_4^- &= 0.010, \\
\Delta q_5^+ = \Delta q_5^- &= 0.005.
\end{aligned}$$

The zero exclusion formulation is:

$$\min\ k$$

$$\begin{aligned}
\text{subject to} \quad a_4(\boldsymbol{q})\omega^4 - a_2(\boldsymbol{q})\omega^2 + a_0(\boldsymbol{q}) &= 0 \\
a_3(\boldsymbol{q})\omega^2 - a_1(\boldsymbol{q}) &= 0
\end{aligned}$$

$$\begin{aligned}
10.0 - 1.0k &\le q_1 \le 10.0 + 1.0k \\
1.0 - 0.1k &\le q_2 \le 1.0 + 0.1k \\
1.0 - 0.1k &\le q_3 \le 1.0 + 0.1k \\
0.2 - 0.01k &\le q_4 \le 0.2 + 0.01k \\
0.05 - 0.005k &\le q_5 \le 0.05 + 0.005k.
\end{aligned}$$

The stability margin is found to be $k_m = 6.2746$ and therefore the system is stable. The critical parameter values are :

$$\begin{aligned}
q_1^* &= 16.2746 \\
q_2^* &= 1.6275 \\
q_3^* &= 1.6275 \\
q_4^* &= 0.1373 \\
q_5^* &= 0.0186 \\
\omega^* &= 0.9864.
\end{aligned}$$

Computational requirements were 3076 iterations and 486 seconds of CPU time.

9.2.5 Example 5

This example addresses the stability of a wire guided Daimler Benz 0305 bus (Ackermann *et al.*, 1991). The transfer function of the linearized system is

$$G(s, q_1, q_2) = \frac{q_1\left(48280q_1 + 388600s + 609.8q_1q_2s^2\right)}{\left(16.8q_1^2q_2 + 270000 + 1077q_1q_2s + q_1^2q_2^2s^2\right)s^3}.$$

The transfer function of the controller is

$$C(s) = \frac{9375 + 10938s + 2344s^2}{15625 + 1250s + 50s^2 + s^3}.$$

The closed–loop characteristic polynomial is

$$P(s, q_1, q_2) = \sum_{i=0}^{8} a_i(q_1, q_2)s^i$$

$$\begin{aligned}
\text{where } a_0(q_1,q_2) &= 453\,10^6q_1^2 \\
a_1(q_1,q_2) &= 528\,10^6q_1^2 + 3640\,10^6q_1 \\
a_2(q_1,q_2) &= 5.72\,10^6q_1^2q_2 + 113\,10^6q_1^2 + 4250\,10^6q_1 \\
a_3(q_1,q_2) &= 6.93\,10^6q_1^2q_2 + 911\,10^6q_1 + 4220\,10^6 \\
a_4(q_1,q_2) &= 1.45\,10^6q_1^2q_2 + 16.8\,10^6q_1q_2 + 338\,10^6 \\
a_5(q_1,q_2) &= 15.6\,10^3q_1^2q_2^2 + 840q_1^2q_2 + 1.35\,10^6q_1q_2 + 13.5\,10^6 \\
a_6(q_1,q_2) &= 1.25\,10^3q_1^2q_2^2 + 16.8q_1^2q_2 + 53.9\,10^3q_1q_2 + 270\,10^3 \\
a_7(q_1,q_2) &= 50q_1^2q_2^2 + 1080q_1q_2 \\
a_8(q_1,q_2) &= q_1^2q_2^2.
\end{aligned}$$

The nominal parameter values are

$$q_1^N = 17.5,\ q_2^N = 20.0$$

and the parameter perturbations

$$\Delta q_1^+ = \Delta q_1^- = 14.5,\ \Delta q_2^+ = \Delta q_2^- = 15.$$

The zero exclusion formulation is:

$$\min\ k$$

$$\begin{aligned}
\text{subject to}\quad a_8(\boldsymbol{q})\omega^8 - a_6(\boldsymbol{q})\omega^6 + a_4(\boldsymbol{q})\omega^4 - a_2(\boldsymbol{q})\omega^2 + a_0(\boldsymbol{q}) &= 0 \\
a_7(\boldsymbol{q})\omega^6 - a_5(\boldsymbol{q})\omega^4 + a_3(\boldsymbol{q})\omega^2 - a_1(\boldsymbol{q}) &= 0
\end{aligned}$$

$$\begin{aligned}
17.5 - 14.5k &\le q_1 \le 17.5 + 14.5k \\
20.0 - 15.0k &\le q_2 \le 20.0 + 15.0k.
\end{aligned}$$

The system was found to be stable, $k_m > 1.0$, for the given range of uncertain parameters. Computational requirements were 109 iterations and 22 seconds of CPU time.

9.2.6 Example 6

In this example, an analysis of the stability margin of the spark ignition engine Fiat Dedra (Barmish (1994), Abate *et al.* (1994)) is carried out for a continuous range of operating conditions involving seven uncertain parameters. The closed–loop characteristic polynomial of seventh degree is:

$$P(s, \boldsymbol{q}) = \sum_{i=0}^{7} a_i(\boldsymbol{q}) s^i$$

where the characteristic polynomial coefficients $a_i(\boldsymbol{q})$, $i = 0, \ldots, 7$ are polynomial functions of q_i, $i = 0, \ldots, 7$ and take the form :

$$\begin{aligned}
a_7(\boldsymbol{q}) &= q_7^2 \\
a_6(\boldsymbol{q}) &= 0.1586 q_1 q_7^2 + 2 q_2 q_7^2 + 2 q_5 q_7 + 0.1826 q_6 q_7 + 0.0552 q_7^2 \\
a_5(\boldsymbol{q}) &= 0.0189477 q_1 q_7^2 + 0.11104 q_2 q_7^2 + 0.1826 q_5 q_6 + 0.1104 q_5 q_7 \\
&\quad + 0.0237398 q_6 q_7 + q_2^2 q_7^2 + 0.1586 q_1 q_2 q_7^2 + 0.0872 q_1 q_4 q_7 \\
&\quad + 0.0215658 q_1 q_6 q_7 + 0.3652 q_2 q_6 q_7 + 2 q_3 q_4 q_7 - 0.0848 q_3 q_6 q_7 \\
&\quad + q_5^2 + 7.61760\ 10^{-4} q_7^2 + 0.3172 q_1 q_5 q_7 + 4 q_2 q_5 q_7 \\
a_4(\boldsymbol{q}) &= 0.1586 q_1 q_5^2 + 4.02141\ 10^{-4} q_1 q_7^2 + 2 q_2 q_5^2 + 0.00152352 q_2 q_7^2 \\
&\quad + 0.0237398 q_5 q_6 + 0.00152352 q_5 q_7 + 5.16120\ 10^{-4} q_6 q_7 \\
&\quad + 0.0552 q_2^2 q_7^2 + 0.01898477 q_1 q_2 q_7^2 + 0.0872 q_1 q_4 q_5 \\
&\quad + 0.034862 q_1 q_4 q_7 + 0.0215658 q_1 q_5 q_6 + 0.00287416 q_1 q_6 q_7 \\
&\quad + 0.0474795 q_2 q_6 q_7 + 2 q_3 q_4 q_5 + 0.1826 q_3 q_4 q_6 + 0.1104 q_3 q_4 q_7 \\
&\quad - 0.0848 q_3 q_5 q_6 - 0.00234048 q_3 q_6 q_7 + 2 q_2^2 q_5 q_7 + 0.1826 q_2^2 q_6 q_7 \\
&\quad + 0.0872 q_1 q_2 q_4 q_7 + 0.3172 q_1 q_2 q_5 q_7 + 0.0215658 q_1 q_2 q_6 q_7 \\
&\quad + 0.1586 q_1 q_3 q_4 q_7 + 2 q_2 q_3 q_4 q_7 - 0.0848 q_2 q_3 q_6 q_7 \\
&\quad + 0.0552 q_5^2 + 0.3652 q_2 q_5 q_6 + 0.0378954 q_1 q_5 q_7 \\
&\quad + 0.2208 q_2 q_5 q_7
\end{aligned}$$

$$
\begin{aligned}
a_3(\boldsymbol{q}) \;=\;& 0.0189477 q_1 q_5^2 + 0.1104 q_2 q_5^2 + 5.16120\ 10^{-4} q_5 q_6 + q_2^2 q_5^2 \\
&+7.61760\ 10^{-4} q_2^2 q_7^2 + q_3^2 q_4^2 + 0.1586 q_1 q_2 q_5^2 + 4.02141\ 10^{-4} q_1 q_2 q_7^2 \\
&+0.0872 q_1 q_3 q_4^2 + 0.034862 q_1 q_4 q_5 + 0.00336706 q_1 q_4 q_7 \\
&+0.00287416 q_1 q_5 q_6 + 6.28987\ 10^{-5} q_1 q_6 q_7 + 0.00103224 q_2 q_6 q_7 \\
&+0.1104 q_3 q_4 q_5 + 0.0237398 q_3 q_4 q_6 + 0.00152352 q_3 q_4 q_7 \\
&-0.00234048 q_3 q_5 q_6 + 0.1826 q_2^2 q_5 q_6 + 0.1104 q_2^2 q_5 q_7 \\
&+0.0237398 q_2^2 q_6 q_7 - 0.0848 q_3^2 q_4 q_6 + 0.0872 q_1 q_2 q_4 q_5 \\
&+0.034862 q_1 q_2 q_4 q_7 + 0.0215658 q_1 q_2 q_5 q_6 + 0.0378954 q_1 q_2 q_5 q_7 \\
&+0.00287416 q_1 q_2 q_6 q_7 + 0.1586 q_1 q_3 q_4 q_5 + 0.0215658 q_1 q_3 q_4 q_6 \\
&+0.0189477 q_1 q_3 q_4 q_7 + 2 q_2 q_3 q_4 q_5 + 0.1826 q_2 q_3 q_4 q_6 \\
&+0.1104 q_2 q_3 q_4 q_7 - 0.0848 q_2 q_3 q_5 q_6 - 0.00234048 q_2 q_3 q_6 q_7 \\
&+7.61760\ 10^{-4} q_5^2 + 0.0474795 q_2 q_5 q_6 + 8.04282\ 10^{-4} q_1 q_5 q_7 \\
&+0.00304704 q_2 q_5 q_7
\end{aligned}
$$

$$
\begin{aligned}
a_2(\boldsymbol{q}) \;=\;& 4.02141\ 10^{-4} q_1 q_5^2 + 0.00152352 q_2 q_5^2 + 0.0552 q_2^2 q_5^2 + 0.0552 q_3^2 q_4^2 \\
&+0.0189477 q_1 q_2 q_5^2 + 0.034862 q_1 q_3 q_4^2 + 0.00336706 q_1 q_4 q_5 \\
&+6.82079\ 10^{-5} q_1 q_4 q_7 + 6.28987\ 10^{-5} q_1 q_5 q_6 + 0.00152352 q_3 q_4 q_5 \\
&+5.16120\ 10^{-4} q_3 q_4 q_6 - 0.00234048 q_3^2 q_4 q_6 + 0.034862 q_1 q_2 q_4 q_5 \\
&+0.0237398 q_2^2 q_5 q_6 + 0.00152352 q_2^2 q_5 q_7 + 5.16120\ 10^{-4} q_2^2 q_6 q_7 \\
&+0.00336706 q_1 q_2 q_4 q_7 + 0.00287416 q_1 q_2 q_5 q_6 \\
&+8.04282\ 10^{-4} q_1 q_2 q_5 q_7 + 6.28987\ 10^{-5} q_1 q_2 q_6 q_7 \\
&+0.0189477 q_1 q_3 q_4 q_5 + 0.00287416 q_1 q_3 q_4 q_6 \\
&+4.02141\ 10^{-4} q_1 q_3 q_4 q_7 + 0.1104 q_2 q_3 q_4 q_5 \\
&+0.0237398 q_2 q_3 q_4 q_6 + 0.00152352 q_2 q_3 q_4 q_7 \\
&-0.00234048 q_2 q_3 q_5 q_6 + 0.00103224 q_2 q_5 q_6
\end{aligned}
$$

$$
\begin{aligned}
a_1(\boldsymbol{q}) \;=\;& 7.61760\ 10^{-4} q_2^2 q_5^2 + 7.61760\ 10^{-4} q_3^2 q_4^2 + 4.02141\ 10^{-4} q_1 q_2 q_5^2 \\
&+0.00336706 q_1 q_3 q_4^2 + 6.82079\ 10^{-5} q_1 q_4 q_5 + 5.16120\ 10^{-4} q_2^2 q_5 q_6 \\
&+0.00336706 q_1 q_2 q_4 q_5 + 6.82079\ 10^{-5} q_1 q_2 q_4 q_7 \\
&+6.28987\ 10^{-5} q_1 q_2 q_5 q_6 + 4.02141\ 10^{-4} q_1 q_3 q_4 q_5 \\
&+6.28987\ 10^{-5} q_1 q_3 q_4 q_6 + 0.00152352 q_2 q_3 q_4 q_5 \\
&+5.16120\ 10^{-4} q_2 q_3 q_4 q_6
\end{aligned}
$$

$$
a_0(\boldsymbol{q}) \;=\; 6.82079\ 10^{-5} q_1 q_3 q_4^2 + 6.82079\ 10^{-5} q_1 q_2 q_4 q_5 .
$$

The nominal parameter values and perturbations are

$$
\begin{array}{lllllll}
q_1^N & = & 3.4329, & \Delta q_1^+ & = & 0, & \Delta q_1^- = 1.2721 \\
q_2^N & = & 0.1627, & \Delta q_2^+ & = & 0, & \Delta q_2^- = 0.06 \\
q_3^N & = & 0.1139, & \Delta q_3^+ & = & 0, & \Delta q_3^- = 0.0782 \\
q_4^N & = & 0.2539, & \Delta q_4^+ & = & 0.3068, & \Delta q_4^- = 0 \\
q_5^N & = & 0.0208, & \Delta q_5^+ & = & 0, & \Delta q_5^- = 0.0108 \\
q_6^N & = & 2.0247, & \Delta q_6^+ & = & 2.4715, & \Delta q_6^- = 0 \\
q_7^N & = & 1.0000, & \Delta q_7^+ & = & 9.0000, & \Delta q_7^- = 0.
\end{array}
$$

The zero exclusion formulation yields:

$$\min \; k$$

$$
\begin{aligned}
\text{subject to} \quad -a_6(\boldsymbol{q})\omega^6 + a_4(\boldsymbol{q})\omega^4 + -a_2(\boldsymbol{q})\omega^2 + a_0(\boldsymbol{q}) &= 0 \\
a_7(\boldsymbol{q})\omega^6 - a_5(\boldsymbol{q})\omega^4 + a_3(\boldsymbol{q})\omega^2 - a_1(\boldsymbol{q}) &= 0
\end{aligned}
$$

$$
\begin{array}{rcl}
3.4329 - 1.2721k & \leq q_1 \leq & 3.4329 \\
0.1627 - 0.06k & \leq q_2 \leq & 0.1627 \\
0.1139 - 0.0782k & \leq q_3 \leq & 0.1139 \\
0.2539 & \leq q_4 \leq & 0.2539 + 0.3068k \\
0.0208 - 0.0108k & \leq q_5 \leq & 0.0208 \\
2.0247 & \leq q_6 \leq & 2.0247 + 2.4715k \\
1.0000 & \leq q_7 \leq & 1.0000 + 9.0000k.
\end{array}
$$

The system was found to be stable, $k_m > 1.0$, throughout the entire range of the uncertain parameters. Computational requirements were 4,896 iterations and 10,000 seconds of CPU time.

Summary

In this chapter, we have discussed the application of the global optimization approach presented in Chapter 8 to a large number of engineering design and control problems that can be modeled as generalized geometric programming *(GGP)* problems. In all cases, convergence to the global minimum was achieved with reasonable computational effort. For additional computational studies and test problems the reader is directed to Maranas and Floudas (1997) and Floudas *et al.* (1999).

III Towards General Twice Differentiable NLPs

10 FROM BICONVEX TO GENERAL TWICE DIFFERENTIABLE NLPS

In this chapter we present an important theoretical development (Liu and Floudas, 1993) that allows us to extend the GOP global optimization approach from biconvex and bilinear problems into general constrained twice-differentiable nonlinear optimization problems. This theoretical development provides the link between the GOP algorithm and the αBB global optimization approach, which is our focus on Part III of this book. Section 10.1 presents the theoretical developments that lead from the GOP to the αBB global optimization approach. Section 10.2 presents a brief introduction to the calculation of the α parameters and one alternative for obtaining rigorous bounds on the α parameters for general constrained twice differentiable nonlinear optimization problems. The material presented in this chapter is based on the work of Liu and Floudas (1993) and Maranas and Floudas (1994b).

10.1 THEORETICAL RESULTS : GOP TO αBB

In this section, we focus on the general constrained twice differentiable nonlinear optimization problems stated as :

C. A. Floudas, *Deterministic Global Optimization*
© Springer Science+Business Media Dordrecht 2000

$$\begin{aligned} \min_{\boldsymbol{x}} \quad & F(\boldsymbol{x}) \\ s.t. \quad & G_j(\boldsymbol{x}) \leq 0 \quad j = 1, \ldots, J \\ & \boldsymbol{x} \in X \subseteq \Re^n \end{aligned} \tag{10.1}$$

where F and G belong to C^2, the set of continuous twice-differentiable functions, $\boldsymbol{x}$ is a vector of size n, and X is a non-empty compact convex set. This formulation represents a very large class of optimization problems since it only assumes that the functions belong to C^2. Hence, arbitrary nonlinear functions such as those that may have trigonometric terms or signomial terms are included in the above formulation. Note also that equality constraints are included by writing them as two inequalities with less than or equal and greater than or equal signs.

Lemma 10.1.1 *Let $A(\boldsymbol{x})$ be a continuous symmetric $(n \times n)$ matrix on X, and I is the identity $(n \times n)$ unit matrix. Then, there exists a parameter $\alpha_0 > 0$ such that the matrix $B(\boldsymbol{x})$ defined as :*

$$B(\boldsymbol{x}) = A(\boldsymbol{x}) + \alpha I \tag{10.2}$$

is a positive definite matrix on X for $\alpha \geq \alpha_0$.

Proof. Dividing (10.2) by α, we note that

$$\frac{B(\boldsymbol{x})}{\alpha} = \frac{A(\boldsymbol{x})}{\alpha} + I \tag{10.3}$$

and that the infinity norm of $||\frac{A(\boldsymbol{x})}{\alpha}|| \to 0$ as $\alpha \to \infty$. As a result, the determinant and the subdeterminants of $\frac{B(\boldsymbol{x})}{\alpha}$ will converge to the corresponding values of I as $\alpha \to \infty$. Hence, there is an $\alpha_0 > 0$ such that $\frac{B(\boldsymbol{x})}{\alpha}$ is positive definite for $\alpha \geq \alpha_0$, and so is the case for $B(\boldsymbol{x})$.

With this lemma as a basis, we have the following important theorem :

Theorem 10.1 Let X, $F(\boldsymbol{x})$, $G_j(\boldsymbol{x})$ for $j = 1, \ldots, J$ satisfy the conditions of (10.1). Then, there exist C^2 functions $f(\boldsymbol{x}, \boldsymbol{y})$, $h_m(\boldsymbol{x}, \boldsymbol{y})$, $g_j(\boldsymbol{x}, \boldsymbol{y})$ for $m = 1, \ldots, M$, $j = 1, \ldots, J$ that satisfy the *Conditions (A)* of the GOP global optimization approach, and therefore (10.1) can be reformulated in the standard

GOP form :

$$\begin{aligned} \min_{\boldsymbol{x},\boldsymbol{y}} \quad & f(\boldsymbol{x},\boldsymbol{y}) \\ s.t. \quad h_m(\boldsymbol{x},\boldsymbol{y}) &= 0 \quad m = 1,\ldots,M \\ g_j(\boldsymbol{x},\boldsymbol{y}) &\leq 0 \quad j = 1,\ldots,J \\ \boldsymbol{x} &\in X \\ \boldsymbol{y} &\in Y. \end{aligned} \tag{10.4}$$

Proof. Based on the above lemma, we select an $\alpha > 0$ such that the following matrices :

$$H_F(\boldsymbol{x}) + 2\alpha I \tag{10.5}$$

$$H_{G_j}(\boldsymbol{x}) + 2\alpha I \qquad j = 1,\ldots,J \tag{10.6}$$

are positive definite matrices on X, where $H_F(\boldsymbol{x})$ and $H_{G_j}(\boldsymbol{x})$ are the Hessian matrices of the functions $F(\boldsymbol{x})$ and $G_j(\boldsymbol{x})$ for $j = 1,\ldots,J$ defined on X. If we now define the functions $f(\boldsymbol{x},\boldsymbol{y})$, $h_i(\boldsymbol{x},\boldsymbol{y})$, and $g_j(\boldsymbol{x},\boldsymbol{y})$ such that :

$$f(\boldsymbol{x},\boldsymbol{y}) = F(\boldsymbol{x}) + \alpha\boldsymbol{x}\boldsymbol{x}^T - \alpha\boldsymbol{x}\boldsymbol{y}^T \tag{10.7}$$

$$h_m(\boldsymbol{x},\boldsymbol{y}) = x_i - y_i \quad i = 1,\ldots,n \quad m = 1,\ldots,M \tag{10.8}$$

$$g_j(\boldsymbol{x},\boldsymbol{y}) = G_j(\boldsymbol{x}) + \alpha\boldsymbol{x}\boldsymbol{x}^T - \alpha\boldsymbol{x}\boldsymbol{y}^T \quad j = 1,\ldots,J \tag{10.9}$$

then, it follows that problem (10.1) is written in the standard GOP form where the *Conditions (A)* are satisfied.

Remark 10.1.1 It is important to select the parameters α properly so as to result in positive definite Hessians. This can be addressed by noting that the Hessian matrices $H_F(\boldsymbol{x})$ and $H_{G_j}(\boldsymbol{x})$ can be decomposed as QDQ^T and $Q_iD_iQ_i^T$ where the matrices D and D_i are diagonal matrices whose elements are the eigenvalues of $H_F(\boldsymbol{x})$ and $H_{G_j}(\boldsymbol{x})$ and $QQ^T = Q_iQ_i^T = I$. Then, it follows that the α parameter can be chosen as :

$$\alpha = -\frac{1}{2}\min_{\boldsymbol{x}\in X}(0, \lambda(\boldsymbol{x}), \lambda_1(\boldsymbol{x}), \ldots, \lambda_J(\boldsymbol{x})) \tag{10.10}$$

where $\lambda(\boldsymbol{x})$ and $\lambda_1(\boldsymbol{x}),\ldots,\lambda_J(\boldsymbol{x})$ are the minimum eigenvalues of the Hessian of the objective function, $H_F(\boldsymbol{x})$, and the Hessians of the inequality constraints, $H_{G_j}(\boldsymbol{x})$, for $j = 1,\ldots,J$ over the domain $x \in X$.

10.2 ESTIMATION OF THE PARAMETER α

In this section, we briefly outline the key issues about the determination of the parameter α and we discuss one possible way of obtaining valid bounds on α.

In addition to this brief description, Adjiman *et al.* (1998a) presented several rigorous methods for calculating the parameters α for general constrained twice differentiable NLPs. These are discussed in detail in Chapter 12.

It is evident that a lower bound on the eigenvalues suffices so as to achieve the desirable transformation. Hence, we focus on how to obtain rigorous lower bounds on the eigenvalues of the Hessian of the objective function since the same approach can be applied to each of the constraints.

As it has been shown, α is a nonnegative parameter and if H_F is to be made positive semidefinite then α must be at least equal to minus one half of its minimum eigenvalue. Therefore, ideally one would want the parameter α to be exactly equal to minus one half of the minimum eigenvalue of H_F :

$$\alpha = \max\left\{0, -\frac{1}{2}\min_{\substack{i \\ \boldsymbol{x}\in X}}(\lambda_i(\boldsymbol{x}))\right\}.$$

However, explicit expressions for the eigenvalues λ_i as functions of $\boldsymbol{x}$, in general, can not be derived. Therefore, the minimum eigenvalue $\lambda_i(\boldsymbol{x})$ both over i and $\boldsymbol{x}$ can implicitly be obtained with the solution of the following optimization problem,

$$\min_{\boldsymbol{x}\in X} \lambda$$

$$\text{subject to} \qquad \det\left[H_F(\boldsymbol{x}) - \lambda I\right] = 0$$

where $H_F(\boldsymbol{x})$ is the Hessian matrix of F as a function of $\boldsymbol{x}$, and I the identity matrix. The equality constraint in the above formulation involves n solutions for λ. Each solution $\lambda(\boldsymbol{x})$ must then be minimized over $\boldsymbol{x} \in X$, and the negative one half of this solution is the parameter α. Because the lengthy expansion of the determinant corresponds to a nonconvex equality constraint, solutions of the above problem can be guaranteed to correspond to only local minima. However, by repetitively solving this problem from different initial points one increases the chances of locating the global minimum. Even if a mere local minimum is obtained, a safety margin can be used to compensate for the difference from the global minimum. In principle, however, only lower bounds on the required value of the parameter α can be obtained with this formulation.

Valid bounds on α can be derived by using the concept of a measure of a matrix. The measure $\mu(A)$ of a square matrix A is a mapping from $R^{n\times n}$ into R (see, for instance, Kokossis and Floudas (1994a), and Maranas and Floudas (1994b)),

$$\mu(A) = \lim_{\epsilon\to 0^+} \frac{\| I + \epsilon A \| - 1}{\epsilon}$$

defined as the directional derivative of the induced norm $\| I + \epsilon A \|$ at the point I in the direction of A. By definition, the induced norm of a matrix A

corresponds to the largest amount by which any vector in R^n is amplified when multiplied by A. Different induced norms give rise to different expressions for the measure of a matrix A :

$$\begin{array}{lclcrcl} \|A\|_1 & = & \max\limits_j \sum\limits_i |a_{ij}| & \longrightarrow & \mu_1(A) & = & \max\limits_j \left[a_{jj} + \sum\limits_{j \neq i} |a_{ij}| \right] \\ \|A\|_2 & = & \max\limits_i \lambda_i \left[(A^* A)^{1/2} \right] & \longrightarrow & \mu_2(A) & = & \max\limits_i \lambda_i \left(\frac{A^* + A}{2} \right) \\ \|A\|_\infty & = & \max\limits_i \sum\limits_j |a_{ij}| & \longrightarrow & \mu_\infty(A) & = & \max\limits_i \left[a_{ii} + \sum\limits_{i \neq j} |a_{ij}| \right]. \end{array}$$

The measure of a matrix has been shown to satisfy a number of important properties. Among these properties are the following three:

Property 10.2.1 *The measure $\mu(A)$ of a square matrix A always exists.*

Property 10.2.2 $\mu(aA+(1-a)B) \leq a\mu(A)+(1-a)\mu(B), \quad \forall\ a \in [0,1].$

Property 10.2.3 $-\mu(-A) \leq \text{Real}\{\lambda_i\} \leq \mu(A), \quad i = 1, \ldots, n.$

These properties imply that the measure $\mu(A)$ of a matrix A is always uniquely defined, is convex, and can be used for deriving lower and upper bounds on the eigenvalues of matrix A. Based on these properties, an upper bound for the parameter α can be obtained by solving the following optimization problem.

$$\alpha = -\frac{1}{2} \min_{\boldsymbol{x} \in X} -\mu\left(-H_F(\boldsymbol{x})\right) = \frac{1}{2} \max_{\boldsymbol{x} \in X} \mu\left(-H_F(\boldsymbol{x})\right)$$

where $H_F(\boldsymbol{x})$ is the Hessian matrix of F as a function of $\boldsymbol{x}$. The next step is to select which measure $\mu(1, 2 \text{ or } \infty)$ will be employed in the optimization problem. Clearly, μ_1 and μ_∞ are more suitable because the maximization terms they involve can be easily reduced to a set of inequality constraints. Furthermore, it is quite straightforward to show that because $H_F(\boldsymbol{x})$ is a symmetric matrix, the expressions for μ_1 and μ_∞ are identical and that the eigenvalues of $H_F(\boldsymbol{x})$ are all real. Based on the definition of the measure of a matrix, an upper bound on α is given by the following formulation,

$$\begin{array}{rcl} \alpha & = & \min\limits_{\boldsymbol{x} \in X, m \geq 0} \dfrac{m}{2} \\ m & \geq & h_{ii} + \sum\limits_{\substack{i'=1 \\ i' \neq i}}^{n} |h_{ii'}(\boldsymbol{x})|, \quad i = 1, \ldots, n \end{array}$$

where $h_{ii'}(\boldsymbol{x})$ is the $(i, i')th$ element of the Hessian matrix,

$$h_{ii'} = \frac{\partial^2 F}{\partial x_i \partial x_{i'}}.$$

Based on the implicit minimization of the minimum eigenvalue of $F(\boldsymbol{x})$ and the maximization of the measure of the negative Hessian matrix of $F(\boldsymbol{x})$ respectively, lower and upper bounds on the parameter α can be obtained.

Summary

In this chapter, we discussed the fundamental theoretical properties that allow us to move from biconvex and bilinear problems to general constrained twice differentiable NLPs. These are based on a novel (D.C.) transformation which forms the basis for the αBB global optimization approach. An outline of a method for obtaining valid bounds on the α parameters is discussed. For detailed description and analysis of several rigorous methods for the calculation of the α parameters the reader is directed to Chapter 12.

11 THE αBB APPROACH FOR BOX CONSTRAINED TWICE-DIFFERENTIABLE NLPS : THEORY

In this chapter, we discuss the theory of the αBB for box constrained twice-differentiable nonlinear optimization problems. Section 11.1 presents the formulation of the box constrained nonlinear problems, and the transformation into a D.C. problem of special structure that leads into the *Conditions (A)* required by the GOP. Section 11.2 presents the important derivation of a convex lower bounding function, $\mathcal{L}$, based on duality theory, and discusses all the theoretical properties that $\mathcal{L}$ satisfies. Section 11.3 introduces the αBB global optimization approach for box constrained twice-differentiable NLPs and presents its convergence proof. Section 11.4 presents the complexity analysis of the αBB approach. The material presented in this chapter is based on the work of Maranas and Floudas (1994a), (1994b).

11.1 FORMULATION AND D.C. TRANSFORMATION

The formulation of the box constrained twice-differentiable NLPs is :

$$\begin{aligned} \min_{\boldsymbol{x}} \quad & f(\boldsymbol{x}) \\ s.t. \quad & \boldsymbol{x}^L \leq \boldsymbol{x} \leq \boldsymbol{x}^U \\ & \boldsymbol{x} \in \Re^n \end{aligned} \tag{11.1}$$

C. A. Floudas, *Deterministic Global Optimization*
© Springer Science+Business Media Dordrecht 2000

where f belongs to C^2, and $\boldsymbol{x}$ is a vector of size n.

The important ideas in developing a difference of convex functions (D.C.) transformation of the objective function, $f(\boldsymbol{x})$, consist of (i) adding a separable convex quadratic term, (ii) substracting the same term which is rewritten in the form of bilinear terms, $x_i y_i$, (iii) reformulation of the problem with linear constraints of the type $x_i = y_i$, $i = 1 \ldots, n$, so as to be equivalent to (11.1), and (iv) introducing additional restrictions that will provide the connection to the *Conditions (A)* required by the GOP algorithm. These key ideas are illustrated in the following remarks.

The separable convex quadratic form that is added to the objective function $f(\boldsymbol{x})$ is :

$$\alpha \sum_{i=1}^{n} x_i^2 \tag{11.2}$$

where the parameter α is nonnegative.

Remark 11.1.1 Note that in the separable convex quadratic form one parameter α is introduced merely for simplicity of the presentation. We can have one parameter α_i for each of the separable terms in the respective variables x_i. In this case, the separable convex quadratic form that is added is :

$$\sum_{i=1}^{n} \alpha_i x_i^2. \tag{11.3}$$

The subtraction of the term written in bilinear form leads to the equivalent reformulation of the problem :

$$\begin{aligned} \min_{\boldsymbol{x},\boldsymbol{y}} \quad & f(\boldsymbol{x}) + \alpha \textstyle\sum_{i=1}^{n} x_i^2 - \alpha \sum_{i=1}^{n} x_i y_i \\ s.t. \quad & x_i = y_i & i = 1, \ldots, n \\ & x_i^L \leq x_i, y_i \leq x_i^U & i = 1, \ldots, n \\ & \boldsymbol{x}, \boldsymbol{y} \in \Re^n. \end{aligned} \tag{11.4}$$

Notice that the linear equality constraints of (11.4) result in the cancellation of the summation terms and hence we have the original problem (11.1).

The next step is to define a new function $\phi(\boldsymbol{x})$ that is equal to the sum of the first two terms of the objective function of (11.4) :

$$\phi(\boldsymbol{x}) = f(\boldsymbol{x}) + \alpha \sum_{i=1}^{n} x_i^2, \tag{11.5}$$

and to determine the nonnegative parameter α such that the function $\phi(\boldsymbol{x})$ is convex. One way that this can be achieved is via eigenvalue analysis.

Property 11.1.1 *The function $\phi(\boldsymbol{x})$ is convex if the parameter α satisfies :*

$$\alpha \geq \max\left\{ \max_{\substack{i \\ x_i^L \leq x_i \leq x_i^U}} 0, \left(-\frac{1}{2}\lambda_i^f\right) \right\} \tag{11.6}$$

where λ_i^f are the eigenvalues of the Hessian matrix of the function f.

Remark 11.1.2 This property states that the parameter α has to be nonnegative and greater than or equal to the negative one half of the minimum eigenvalue of the Hessian matrix of $f(\boldsymbol{x})$ for every x_i such that $x_i^L \leq x_i \leq x_i^U$ for $i = 1, \ldots, n$.

Using this property which makes the function $\phi(\boldsymbol{x})$ convex, formulation (11.4) can be written as :

$$\begin{aligned} \min_{\boldsymbol{x},\boldsymbol{y}} \quad & \phi(\boldsymbol{x}) - \alpha \textstyle\sum_{i=1}^n x_i y_i \\ s.t. \quad & x_i = y_i \qquad i = 1, \ldots, n \\ & x_i^L \leq x_i, y_i \leq x_i^U \qquad i = 1, \ldots, n \\ & \boldsymbol{x}, \boldsymbol{y} \in \Re^n. \end{aligned} \tag{11.7}$$

Remark 11.1.3 Note that formulation (11.7) satisfies the *Conditions (A)* of the GOP global optimization approach. In fact, the objective function is convex in $\boldsymbol{x}$ for any fixed $\boldsymbol{y}$ and linear in $\boldsymbol{y}$ for any fixed $\boldsymbol{x}$, while the constraints are linear and the constraint qualification condition is trivially satisfied. Hence, we have been able to establish the relation between box constrained nonlinear twice differentiable optimization problems and the classes of biconvex problems that meet *Conditions (A)* of the GOP algorithm as presented in Chapters 3-7.

Remark 11.1.4 Formulation (11.7) corresponds to a special type of (D.C.) transformation that forms the basis for the αBB global optimization approach. Note also that formulation (11.7) can be equivalently written as :

$$\begin{aligned} \min_{\boldsymbol{x},\boldsymbol{y}} \quad & \phi(\boldsymbol{y}) - \alpha \textstyle\sum_{i=1}^n x_i y_i \\ s.t. \quad & x_i = y_i \qquad i = 1, \ldots, n \\ & x_i^L \leq x_i, y_i \leq x_i^U \qquad i = 1, \ldots, n \\ & \boldsymbol{x}, \boldsymbol{y} \in \Re^n. \end{aligned} \tag{11.8}$$

In this case, the objective function is linear in $\boldsymbol{x}$ for fixed $\boldsymbol{y}$ and convex in $\boldsymbol{y}$ for fixed $\boldsymbol{x}$. This form will be used in the proofs of additional properties.

Maranas and Floudas (1994b) showed that we can further exploit the mathematical structure of (11.7) or (11.8) by using duality theory and proved several important theoretical properties which are discussed in the next section.

11.2 CONVEX LOWER BOUNDING FUNCTION

The next step in the analysis is to derive a convex lower bounding function denoted as $\mathcal{L}$. Maranas and Floudas (1994b) proved the following property that leads to the derivation of the function $\mathcal{L}$:

Property 11.2.1 *The relaxed dual formulation of the minimization problem (11.8), and hence problem (11.1), is equivalent to the minimization of the function $\mathcal{L}(\boldsymbol{x})$ defined as :*

$$\begin{aligned} \mathcal{L}(\boldsymbol{x}) &= f(\boldsymbol{x}) + \alpha \sum_{i=1}^{n} (x_i^L - x_i)(x_i^U - x_i) \\ \text{where} \qquad \alpha &\geq \max \left\{ \max_{\substack{i \\ x_i^L \leq x_i \leq y_i^U}} 0, \left(-\frac{1}{2}\lambda_i^f \right) \right\} \end{aligned}$$

where λ_i^f are the eigenvalues of the Hessian matrix of the function f.

Proof. With the introduction of a new set of variables x_i the minimization of $f(x_i)$ over the box constraints defined by the bounds x_i^L, x_i^U can equivalently be reformulated as follows:

$$\begin{aligned} \min_{x_i^L \leq x_i, y_i \leq x_i^U} \quad & f(\boldsymbol{y}) + \alpha \sum_{1=1}^{n} \left(y_i^2 - x_i y_i \right) \\ \text{subject to} \quad & x_i = y_i, \quad i = 1, \ldots, n \end{aligned}$$

where x_i, y_i correspond to the x–type and y–type variables, as presented in Chapter 3 and proposed by Floudas and Visweswaran (1990), Floudas and Visweswaran (1993), Visweswaran and Floudas (1990) and Visweswaran and Floudas (1993). The exact dual of the above problem will then be,

$$\sup_{\lambda_i} \min_{x_i^L \leq x_i, y_i \leq x_i^U} \quad f(\boldsymbol{y}) + \alpha \sum_{i=1}^{n} \left(y_i^2 - x_i y_i \right) + \sum_{i=1}^{n} \lambda_i \left(x_i - y_i \right).$$

The exact dual can be relaxed by dropping the maximization over λ_i and substituting λ_i's with the values derived from the KKT conditions,

$$\lambda_i = \alpha y_i^c, \quad i = 1, \ldots, n$$

where y_i^c is the current point. The formulation for the relaxed dual then becomes,

$$\min_{x_i^L \le x_i, y_i \le x_i^U} \quad f(\boldsymbol{y}) + \alpha \sum_{i=1}^{n} (x_i - y_i)(y_i^c - y_i).$$

The minimization over x_i is accounted by considering all different combinations of signs for the qualifying constraints $y_i^c - y_i$ for $i = 1, \ldots, n$, which are multiplied by x_i. Specifically,

$$\begin{aligned} x_i &= x_i^L \quad \text{if} \quad \forall\, y_i,\ y_i^c - y_i \ge 0, \implies y_i^c = x_i^U, \quad i = 1, \ldots, n \\ x_i &= x_i^U \quad \text{if} \quad \forall\, y_i,\ y_i^c - y_i \le 0, \implies y_i^c = x_i^L, \quad i = 1, \ldots, n. \end{aligned}$$

Note that both cases collapse to the following minimization problem over y_i,

$$\min_{x_i^L \le y_i \le x_i^U} \quad f(\boldsymbol{y}) + \alpha \sum_{i=1}^{n} (x_i^L - y_i)(x_i^U - y_i)$$

which is identical to the minimization of $\mathcal{L}$.

Remark 11.2.1 Note that α is a nonnegative parameter which must be greater than or equal to the negative one half of the minimum eigenvalue of $f(\boldsymbol{x})$ over $x_i^L \le x_i \le x_i^U$. The effect of adding the term,

$$\alpha \sum_{i=1}^{n} (x_i^L - x_i)(x_i^U - x_i)$$

to f is to make $\mathcal{L}$ convex by overpowering the nonconvexity characteristics of f with the addition of the term 2α to all of its eigenvalues,

$$\lambda_i^{\mathcal{L}} = \lambda_i^f + 2\alpha \quad i = 1, \ldots, n.$$

Here $\lambda_i^{\mathcal{L}}$ and λ_i^f are the ith eigenvalues of the Hessians of $\mathcal{L}$ and f, respectively. The function $\mathcal{L}$ satisfies a number of important properties which will be stated and proven next. These properties will enable us to construct a global optimization algorithm for finding the global minimum of f in the space defined by the variables x_i.

Property 11.2.2 *For every $x_i \in [x_i^L, x_i^U]$ for $i = 1, \ldots, n$, we have $\mathcal{L} \le f$.*

Proof.

$$\begin{aligned}
\forall\ x_i \in \left[x_i^L, x_i^U\right], \qquad & \alpha \geq 0,\ \left(x_i^L - x_i\right) \leq 0,\ \left(x_i^U - x_i\right) \geq 0, \\
\Longrightarrow\ & \alpha(x_i^L - x_i)(x_i^U - x_i) \leq 0 \\
\Longrightarrow\ & \alpha \sum_{n=1}^{n} (x_i^L - x_i)(x_i^U - x_i) \leq 0 \\
\Longrightarrow\ & \mathcal{L} \leq f.
\end{aligned}$$

Remark 11.2.2 $\mathcal{L}$ is always a valid underestimator of f inside the box constraints $\left[x_i^L, x_i^U\right]$ for $i = 1, \ldots, n$.

Property 11.2.3 *For every corner point in* $\left[x_i^L, x_i^U\right]$, *we have* $\mathcal{L} = f$.

Proof. Let x_i^c be a corner point of $\left[x_i^L, x_i^U\right]$, $\quad i = 1, \ldots, n$.

$$\begin{aligned}
\forall\ i = 1, \ldots, n, \qquad & (x_i^L - x_i^c) = 0 \quad \text{or} \quad (x_i^U - x_i^c) = 0, \\
\Longrightarrow\ & \alpha(x_i^L - x_i^c)(x_i^U - x_i^c) = 0, \\
\Longrightarrow\ & \sum_{i=1}^{n} \alpha(x_i^L - x_i^c)(x_i^U - x_i^c) = 0, \\
\Longrightarrow\ & \mathcal{L}(x_i^c) = f(x_i^c)
\end{aligned}$$

Remark 11.2.3 $\mathcal{L}$ matches f at all corner points.

Property 11.2.4 $\mathcal{L}$ *is convex in* $\left[x_i^L, x_i^U\right]$.

Proof. $\mathcal{L}$ is convex in $\left[x_i^L, x_i^U\right]$ if and only if for every $x_i \in \left[x_i^L, x_i^U\right]$, all eigenvalues $\lambda_i^{\mathcal{L}}(\boldsymbol{x})$ of $\mathcal{L}$ are nonnegative. This is equivalent to requiring the minimum eigenvalue of $\mathcal{L}$ over x_i to be nonnegative.

$$\min_{\substack{i \\ x_i^L \leq x_i \leq x_i^U}} \lambda_i^{\mathcal{L}}(\boldsymbol{x}) \geq 0.$$

After substituting $\lambda_i^{\mathcal{L}}(\boldsymbol{x}) = \lambda_i^f(\boldsymbol{x}) + 2\alpha$

and $\alpha \geq \max\left\{0, \max\limits_{\substack{i \\ x_i^L \leq x_i \leq x_i^U}} \left(-\frac{1}{2}\lambda_i^f\right)\right\}$ in the previous inequality we have,

$$\min_{\substack{i \\ x_i^L \leq x_i \leq x_i^U}} \left\{\lambda_i^f(\boldsymbol{x}) + \max\left[0, \max_{\substack{i \\ x_i^L \leq x_i \leq x_i^U}} \left(-\lambda_i^f(\boldsymbol{x})\right)\right]\right\} \geq 0.$$

We can further simplify by combining the two max operators and transforming the resulting single max operator into a single min operator.

$$\min_{\substack{i \\ x_i^L \le x_i \le x_i^U}} \lambda_i^f(\boldsymbol{x}) - \min_{\substack{i \\ x_i^L \le x_i \le x_i^U}} \left[0, \lambda_i^f(\boldsymbol{x})\right] \quad \ge \quad 0.$$

Finally, we combine the two min operators by considering the two cases for the sign of $\lambda_i^f(\boldsymbol{x})$. The resulting inequality is true for every x_i in $\left[x_i^L, x_i^U\right]$:

$$\max\left[0, \min_{\substack{i \\ x_i^L \le x_i \le x_i^U}} \lambda_i^f(\boldsymbol{x})\right] \quad \ge \quad 0.$$

Therefore, $\mathcal{L}(\boldsymbol{x})$ is convex for every x_i in $\left[x_i^L, x_i^U\right]$.

Remark 11.2.4 $\mathcal{L}$ is convex in $\left[x_i^L, x_i^U\right]$ by construction.

Property 11.2.5 $$\max_{x_i^L \le x_i \le x_i^U} (f - \mathcal{L}) \;=\; \tfrac{1}{4}\alpha \sum_{i=1}^{n} \left(x_i^U - x_i^L\right)^2$$

Proof.

$$\begin{aligned} & \max_{x_i^L \le x_i \le x_i^U} (f - \mathcal{L}) \\ = \quad & \max_{x_i^L \le x_i \le x_i^U} -\alpha \sum_{i=1}^{n} (x_i^L - x_i)(x_i^U - x_i) \\ = \quad & \min_{x_i^L \le x_i \le x_i^U} \alpha \sum_{i=1}^{n} (x_i^L - x_i)(x_i^U - x_i) \end{aligned}$$

The last function is convex and separable in x_i. This means that if the single stationary point is inside the box constraints $[x_i^L, x_i^U]$ then it corresponds to the single global minimum (Bazaraa *et al.*, 1993). After applying the stationarity conditions we find that the single stationary point, x_i^{stat} is indeed inside the box constraints,

$$x_i^{stat} \quad = \quad \frac{\left(x_i^L + x_i^U\right)}{2}, \quad i = 1, \ldots, n,$$

and therefore it corresponds to the single global minimum. After substituting the expression for x_i^{stat} in the original function, we have,

$$\max_{x_i^L \le x_i \le x_i^U} (f - \mathcal{L}) \quad = \quad \frac{1}{4}\alpha \sum_{i=1}^{n} \left(x_i^U - x_i^L\right)^2.$$

Remark 11.2.5 The maximum separation between $\mathcal{L}$ and f is proportional to α and to the square of the diagonal of the current box constraints.

Property 11.2.6

$$\begin{aligned}
\forall \quad x_i \in \left[x_i^{L1}, x_i^{U1}\right], \quad \forall \quad & \left[x_i^{L2}, x_i^{U2}\right] \text{ such that } \left[x_i^{L1}, x_i^{U1}\right] \subseteq \left[x_i^{L2}, x_i^{U2}\right], \\
& \mathcal{L}(x_i, \alpha^1, x_i^{L1}, x_i^{U1}) \geq \mathcal{L}(x_i, \alpha^2, x_i^{L2}, x_i^{U2}), \\
& \text{after imposing } \alpha^2 \geq \alpha^1 \text{ in the selection process for } \alpha^1,\ \alpha^2,
\end{aligned}$$

$$\begin{aligned}
\text{where} \qquad \alpha^1 &\geq \max\left\{0, \max_{\substack{i \\ x_i^{L1} \leq x_i \leq y_i^{U1}}} \left(-\frac{1}{2}\lambda_i^f(\boldsymbol{x})\right)\right\} \\
\text{and} \qquad \alpha^2 &\geq \max\left\{0, \max_{\substack{i \\ x_i^{L2} \leq x_i \leq x_i^{U2}}} \left(-\frac{1}{2}\lambda_i^f(\boldsymbol{x})\right)\right\}.
\end{aligned}$$

Proof.

$$\begin{aligned}
\forall \quad x_i \in \left[x_i^{L1}, x_i^{U1}\right], \qquad & \alpha^2 \geq \alpha^1 \geq 0, \\
& \left(x_i^{L2} - x_i\right) \leq \left(x_i^{L1} - x_i\right) \leq 0, \\
& \left(x_i^{U2} - x_i\right) \geq \left(x_i^{U1} - x_i\right) \geq 0 \\
\Longrightarrow \quad & \alpha^1 \sum_{i=1}^{n} (x_i^{L1} - x_i)(x_i^{U1} - x_i) \\
\geq \quad & \alpha^2 \sum_{i=1}^{n} (x_i^{L2} - x_i)(x_i^{U2} - x_i) \\
\Longrightarrow \quad & \mathcal{L}(x_i, \alpha^1, x_i^{L1}, x_i^{U1}) \geq \mathcal{L}(x_i, \alpha^2, x_i^{L2}, x_i^{U2})
\end{aligned}$$

Remark 11.2.6 The underestimators constructed over supersets of the current set are always *less tight* than the underestimator constructed over the current box constraints for every point within the current box constraints.

Hence, we have proven that $\mathcal{L}$ is a *convex lower bounding* function of f. Also, $\mathcal{L}$ *matches* f at all corner points of the box constraints inside which it has been defined. The maximum separation distance of $\mathcal{L}$ from f is *bounded* and *proportional* to α and to the square of the diagonal of the box constraints. Finally, the values of different definitions of $\mathcal{L}$ at any point, if $\mathcal{L}$ is constructed over a tighter box of constraints each time, define a *nondecreasing* sequence. In the next section, based on the aforementioned properties, a branch and bound type global optimization algorithm is introduced for solving the global minimization of f.

11.3 GLOBAL OPTIMIZATION ALGORITHM αBB

Using the properties introduced in the previous section, a global optimization algorithm is proposed for locating the global minimum, f^*, by constructing converging lower and upper bounds on f^*. It is clear that the value of f at any point $\boldsymbol{x}$ is an upper bound on f^*. Lower bounds on f^* within some box constraints are derived by using Properties (11.2.2) and (11.2.4). From Property (11.2.2), $\mathcal{L}$ is a lower bounding function of f. Therefore, its global minimum within some box constraints will always underestimate the global minimum of f within the same box constraints. In addition, from Property (11.2.4) we know that $\mathcal{L}$, unlike f, is convex and therefore it involves a single global minimum which can be found with any local optimization algorithm (e.g., MINOS 5.3 (Murtagh and Saunders, 1983)). An upper bound on f^* can then be obtained by simply calculating f at the global minimum point of $\mathcal{L}$. From Property (11.2.5) we know that the gap between the upper bound and the lower bound will be at most,

$$\left(f^U - f^L\right) \leq \frac{1}{4}\alpha \sum_{i=1}^{n} \left(x_i^U - x_i^L\right)^2$$

where f^U and f^L are the upper and lower bounds on the global minimum f^* inside the current box constraints, $\left[x_i^L, x_i^U\right]$, $i = 1, \ldots, n$.

The next step, after establishing an upper and a lower bound on the global minimum, is to tighten them by using Property (11.2.6). Property (11.2.6) implies that the value of $\mathcal{L}$ at every point, and therefore at its global minimum, is increased by restricting the box constraints within which it has been defined. Tighter box constraints can be attained by partitioning the rectangle that the initial box constraints define into a number of smaller rectangles. One way of partitioning is to successively divide the current rectangle in two subrectangles by halving on the middle point of the longest side of the initial rectangle (bisection). At each iteration the lower bound on f^* is simply the minimum over all the minima of $\mathcal{L}$ in every subrectangle composing the initial rectangle. Therefore, a straightforward (bound improving) way of tightening the lower bound, f^L, is to halve at each iteration, only the subrectangle responsible for the infimum of the minima of $\mathcal{L}$ over all subrectangles. This procedure generates a *nondecreasing* sequence for the lower bound f^L on f^*. Furthermore, we construct a *nonincreasing* sequence for the upper bound f^U by selecting it to be the infimum over all the previously recorded upper bounds. If the global minimum of $\mathcal{L}$ in any subrectangle is greater than the current upper bound f^U we can eliminate this subrectangle because the global minimum of f cannot be located inside it (fathoming step).

Property (11.2.5) answers the question of how small these subrectangles must become before the upper and lower bounds of f inside these subrectangles are

within ϵ. If δ is the diagonal of the subrectangle,

$$\delta = \sqrt{\sum_{i=1}^{n} \left(x_i^U - x_i^L\right)^2}$$

and ϵ is the convergence tolerance, from property (11.2.5) we have the following condition for convergence,

$$\epsilon \geq \frac{1}{4}\,\alpha \sum_{i=1}^{n} \left(x_i^U - x_i^L\right)^2 = \frac{1}{4}\alpha\delta^2 \geq f^U - f^L$$

which means that if the diagonal δ of a subrectangle is,

$$\delta < \sqrt{\frac{4\epsilon}{\alpha}}$$

then ϵ–convergence to the global minimum of f has been achieved.

Remark 11.3.1 It is interesting to note that the required for convergence value of δ is proportional to the square root of ϵ. Note also, that δ is inversely proportional to the squared root of the parameter α reflecting that the smaller the value of α is, the faster the convergence rate becomes.

11.4 STEPS OF THE GLOBAL OPTIMIZATION ALGORITHM αBB

The basic steps of the proposed global optimization algorithm are described in the sequel.

STEP 1 - *Initialization*

A convergence tolerance ϵ is selected, and the iteration counter $Iter$ is set to one. Appropriate global bounds x_i^{LBD}, x_i^{UBD} on x_i are chosen and local bounds $x_i^{L,Iter}, x_i^{U,Iter}$ for the first iteration are set to be equal to the global ones. Lower and upper bounds on the global minimum f^{LBD}, f^{UBD} are initialized to negative and positive infinity, respectively, and an initial current point, $x_i^{c,Iter}$, is randomly selected.

STEP 2 - *Update of Upper Bound f^{UBD}*

f is calculated at the current point $f^{c,Iter}$ and the upper bound f^{UBD} is updated as follows,

$$f^{UBD} = \min\left(f^{UBD}, f(x_i^{c,Iter})\right).$$

STEP 3 - *Partitioning of Current Rectangle*

The current rectangle $\left[x_i^{L,Iter}, x_i^{U,Iter}\right]$, $i = 1, \ldots, n$ is partitioned into the following two rectangles ($r = 1, 2$):

$$\begin{bmatrix} x_1^{L,Iter} & x_1^{U,Iter} \\ \vdots & \vdots \\ x_{l^{Iter}}^{L,Iter} & \frac{\left(x_{l^{Iter}}^{L,Iter} + x_{l^{Iter}}^{U,Iter}\right)}{2} \\ \vdots & \vdots \\ x_n^{L,Iter} & x_n^{U,Iter} \end{bmatrix}, \quad \begin{bmatrix} x_1^{L,Iter} & x_1^{U,Iter} \\ \vdots & \vdots \\ \frac{\left(x_{l^{Iter}}^{L,Iter} + x_{l^{Iter}}^{U,Iter}\right)}{2} & x_{l^{Iter}}^{U,Iter} \\ \vdots & \vdots \\ x_n^{L,Iter} & x_n^{U,Iter} \end{bmatrix}$$

where l^{Iter} corresponds to the variable with the scaled longest side in the initial rectangle,

$$l^{Iter} = \arg\max_i \frac{\left(x_i^{U,Iter} - x_i^{L,Iter}\right)}{\left(x_i^{UBD} - x_i^{LBD}\right)}.$$

STEP 4 - *Solution of Convex Problems in two Subrectangles*

Update the parameter α and solve the following convex nonlinear optimization problem in both subrectangles ($r = 1, 2$) by using a local nonlinear optimization solver (e.g., MINOS 5.4 (Murtagh and Saunders, 1983)).

$$\min_{\boldsymbol{x}} \mathcal{L} = f(\boldsymbol{x}) + \alpha \sum_{i=1}^{n} \left(x_i^{L,Iter} - x_i\right)\left(x_i^{U,Iter} - x_i\right).$$

If a solution, $L_{sol}^{r,Iter}$, is less than the current upper bound,

$$L_{sol}^{r,Iter} \leq f^{UBD}$$

then it is stored along with the value of the variable x_i at the solution point $x_{k,sol}^{r,Iter}$.

STEP 5 - *Update Iteration Counter Iter and Lower Bound f^{LBD}*
The iteration counter is increased by one,

$$Iter \longleftarrow Iter + 1$$

and the lower bound f^{LBD} is updated to the minimum solution over the stored ones from previous iterations. Furthermore, the selected solution is removed from the stored set.

$$\begin{aligned} f^{LBD} &= L_{sol}^{r',Iter'} \\ \text{where} \quad L_{sol}^{r',Iter'} &= \min_{r,I} L_{sol}^{r,I}, \quad r = 1, 2, \quad I = 1, \ldots, Iter - 1. \end{aligned}$$

STEP 6 - *Update Current Point $x_i^{c,Iter}$ and Current Bounds $x_i^{L,Iter}, x_i^{U,Iter}$ on x_i*
The current point is selected to be the solution point of the previously obtained

minimum solution in **STEP 5**,

$$x_i^{c,Iter} = x_{k,sol}^{r',Iter'}, \quad i = 1, \ldots, n$$

and the current rectangle becomes the subrectangle containing the previously obtained solution,

$$\left[x_i^{L,Iter}, x_i^{U,Iter}\right] = \begin{bmatrix} x_1^{L,Iter'} & x_1^{U,Iter'} \\ \vdots & \vdots \\ x_{l^{Iter'}}^{L,Iter'} & \dfrac{\left(x_{l^{Iter'}}^{L,Iter'} + x_{l^{Iter'}}^{U,Iter'}\right)}{2} \\ \vdots & \vdots \\ x_n^{L,Iter'} & x_n^{U,Iter'} \end{bmatrix}, \quad \text{if} \quad r' = 1$$

$$\left[x_i^{L,Iter}, x_i^{U,Iter}\right] = \begin{bmatrix} x_1^{L,Iter'} & x_1^{U,Iter'} \\ \vdots & \vdots \\ \dfrac{\left(x_{l^{Iter'}}^{L,Iter'} + x_{l^{Iter'}}^{U,Iter'}\right)}{2} & x_{l^{Iter'}}^{U,Iter'} \\ \vdots & \vdots \\ x_n^{L,Iter'} & x_n^{U,Iter'} \end{bmatrix}, \quad \text{if} \quad r' = 2.$$

STEP 7 - *Check for Convergence*

If $\left(f^{UBD} - f^{LBD}\right) > \epsilon$, then return to **STEP 2** .

Otherwise, ϵ–convergence has been reached and the global minimum solution and solution point are:

$$\begin{aligned} f^* &\longleftarrow f^{U,Iter''} \\ x_i^* &\longleftarrow x_i^{c,Iter''}, \quad i = 1, \ldots, n \\ \text{where} \quad Iter'' &= \arg_I \left\{f^{U,I} = f^{UBD}\right\}, \quad I = 1, \ldots, Iter. \end{aligned}$$

In the following section, a mathematical proof that the proposed global optimization algorithm converges to the the global minimum is given using the analysis of branch and bound global optimization algorithms presented in Horst and Tuy (1990).

11.5 PROOF OF CONVERGENCE TO THE GLOBAL MINIMUM

Convergence properties of a global optimization algorithm depend on:

(i) the limit behavior of the difference $f^U - f^L$ for unfathomed successively refined partitions,

(ii) the subdivision process of the current partitions, and

(iii) the employed selection process of the partition(s) that have to be further refined.

In the αBB global optimization algorithm a lower bound, $f^L_{r,Iter}$, or every partition element $(r, Iter)$ is obtained as the solution of the convex minimization of $\mathcal{L}$ inside the box constraints of the current partition element, and a corresponding upper bound, $f^U_{r,Iter}$, is provided by the value of f at the minimum point of $\mathcal{L}$. The partition element involving the minimum lower bound, $f^L_{r,Iter}$, is selected for further refining according to the bisection rule, and partition elements whose lower bound, $f^L_{r,Iter}$, is greater than the current upper bound, f^U_{Iter}, are fathomed. A sufficient condition for a global optimization algorithm to be convergent to the global minimum, as stated in Horst and Tuy (1990), requires that the bounding operation must be *consistent* and the selection operation must be *bound improving.*

Definition 11.5.1 *A bounding operation is called* **consistent** *if at every step any unfathomed partition can be further refined, and if any infinitely decreasing sequence of successively refined partition elements satisfies,*

$$\lim_{Iter\to\infty} \left(f^U_{Iter} - f^L_{r,Iter}\right) = 0,$$

where $f^L_{r,Iter}$ is the lower bound of f inside the $(r, Iter)$ partition element and f^U_{Iter} is the best upper bound at iteration $Iter$ not necessarily occurring inside the $(r, Iter)$ partition element.

Any partition element $(r^o, Iter^o)$ that is not fathomed eventually becomes the one involving the minimum current lower bound f^L_{iter} and is further refined according to the bisection rule. Property (11.2.6) implies that the subdivisions $(r, Iter)$ of this partition element $(r^o, Iter^o)$ involve lower bounds $f^L_{r,Iter}$ which are at least as tight as $f^L_{r^o,Iter^o}$, that is,

$$\min_r f^L_{r,Iter} \geq f^L_{r^o,Iter^o},$$

which implies that any infinitely refined partition sequence involves infinitely nondecreasing lower bounds.

In practice, the requirement $\lim_{Iter\to\infty} \left(f^U_{Iter} - f^L_{r,Iter}\right) = 0$ for any infinitely decreasing sequence of successively refined partition elements is difficult to verify because f^U_{Iter} is not necessarily attained at the partition element $(r, Iter)$. Therefore, in view of the inequality $f^U_{r,Iter} \geq f^U_{Iter} \geq f^L_{r,Iter}$, the following, at least equally strong condition, suffices:

$$\lim_{Iter\to\infty} \left(f^U_{r,Iter} - f^L_{r,Iter}\right) = 0.$$

From Property (11.2.5) we know that,

$$\frac{1}{4}\alpha\delta^2 \;\geq\; (f^U_{r,Iter} \;-\; f^L_{r,Iter}) \;\geq\; 0.$$

where δ is the diagonal of the current partition element corresponding to a rectangle. Therefore, it suffices to show that

$$\lim_{Iter\to\infty} \frac{1}{4}\alpha\delta^2 \;=\; 0,$$

or equivalently that

$$\lim_{Iter\to\infty} \delta \;=\; 0.$$

Hence, the employed subdivision process for any unfathomed infinitely decreasing sequence of successively refined partition elements is *exhaustive* (Horst and Tuy, 1990).

Definition 11.5.2 *A subdivision process is* exhaustive *if* $\lim_{Iter\to\infty} \delta = 0$ *for all decreasing subsequences of partition elements generated by the subdivision.*

Theorem 11.1 *The subdivision process of the proposed global optimization algorithm is* exhaustive.

Proof. The employed subdivision process is the *bisection* where every partition element $(r, Iter)$ is subdivided into two subrectangles ($r = 1,\ 2$) of equal volume by halving at the midpoint of the longest scaled side. Because all the sides of any partition element are scaled with the sides of the rectangle that the initial global constraints define, the scaled sides of the initial rectangle are all equal to one. Therefore, the condition of always subdividing along the longest side can be satisfied by simply subdividing first along the side $i = 1$, then along the side $i = 2$, etc. until the last side $i = n$ is encountered, at which point the subdivision starts again from $i = 1$. By partitioning in this orderly manner each side of every successively refined partition element is halved *exactly* once every n subdivisions. Consequently, after n subdivisions the diagonal of the resulting partition elements is one half the diagonal of the original partition element. Therefore, as the number of successive subdivisions of a partition element goes to infinity, the diagonals of the resulting partition elements go to zero. This implies that the employed subdivision process is *exhaustive.*

Theorem 11.2 *The bounding operation of the proposed global optimization algorithm is* **consistent**.

Proof. We have shown that the subdivision process is *exhaustive* and therefore,

$$\lim_{Iter\to\infty} \delta \;=\; 0.$$

Furthermore, we have :

$$0 \leq \lim_{\delta \to 0} \left(f^U_{Iter} - f^L_{r,Iter}\right) \leq \lim_{\delta \to 0} \left(f^U_{r,Iter} - f^L_{r,Iter}\right) \leq \lim_{\delta \to 0} \frac{1}{4}\alpha\delta^2 = 0.$$

Hence, for any infinitely decreasing sequence of successively refined partition elements

$$\lim_{Iter \to \infty} \left(f^U_{Iter} - f^L_{r,Iter}\right) = 0,$$

meaning that the employed bounding operation is *consistent.*

Definition 11.5.3 *A selection operation is called* **bound improving** *if at least one partition element where the actual lower bound is attained is selected for further partitioning after a finite number of refinements.*

Theorem 11.3 *The selection operation of the proposed global optimization algorithm is* bound improving.

Proof. The proposed selection operation is clearly *bound improving* because the partition element where the actual lower bound is attained is selected for further partitioning in the immediately following iteration.

Theorem 11.4 *The proposed global optimization algorithm is* convergent *to the global minimum.*

Proof. We have shown that the bounding operation is *consistent* and that the selection operation is *bound improving.* Therefore, according to Theorem IV.3 in Horst and Tuy (1990), the proposed global optimization algorithm is *convergent* to the global minimum.

11.6 COMPLEXITY ANALYSIS

In the previous section, we presented the mathematical proof that the αBB global optimization algorithm converges to an ϵ-global minimum. In practice, however, not mere ϵ–convergence to the global minimum, but convergence in a finite number of iterations is required. In this section, by analyzing the structure (sparsity) of the branch and bound tree resulting from the subdivision process, finite upper and lower bounds on the total number of required iterations for ϵ–convergence are obtained.

Theorem 11.5 *The maximum number of required iterations for ϵ–convergence to the global minimum f^* for the proposed global optimization algorithm is,*

$$I^{max} = \left[\frac{\sum_{i=1}^{n} \left(x_i^{UBD} - x_i^{LBD}\right)^2}{\sqrt{\frac{4\epsilon}{\alpha}}} \right]^n - 1.$$

Proof. The worst-case scanario for convergence occurs when no partition element is fathomed until the last level of the branch and bound tree. In this case, the branch and bound tree is a complete binary tree whose end-nodes correspond to the partition elements of the last iteration. Because ϵ-convergence is achieved only in the last level of the branch and bound tree, the lower and upper bounds of f in all these partition elements are within ϵ. Therefore, the diagonals of all these subrectangles are given by :

$$\delta = \sqrt{\frac{4\epsilon}{\alpha}}.$$

The number of the partition elements corresponding to end-nodes in the branch and bound tree will then be,

$$N_{rec}^{I^{max}} = \left[\frac{\sum_{i=1}^{n}\left(x_i^{UBD} - x_i^{LBD}\right)^2}{\delta}\right]^n = \left[\frac{\sum_{i=1}^{n}\left(x_i^{UBD} - x_i^{LBD}\right)^2}{\sqrt{\frac{4\epsilon}{\alpha}}}\right]^n.$$

The number of levels, NL, in the complete binary branch and bound tree is,

$$NL = \log_2 N_{rec}^{I^{max}} = n\log_2\left[\frac{\sum_{i=1}^{n}\left(x_i^{UBD} - x_i^{LBD}\right)^2}{\sqrt{\frac{4\epsilon}{\alpha}}}\right].$$

The total number of iterations I^{max} is the total number of nodes in the complete binary tree excluding the end nodes. Therefore,

$$\begin{aligned} I^{max} &= \sum_{i=0}^{\#levels-1} 2^i = \frac{2^{\#levels} - 1}{2-1} = 2^{\#levels} - 1 \\ &= \left[\frac{\sum_{i=1}^{n}\left(x_i^{UBD} - x_i^{LBD}\right)^2}{\sqrt{\frac{4\epsilon}{\alpha}}}\right]^n - 1. \end{aligned}$$

By following the same line of reasoning we can determine the minimum total number of required iterations, as stated in the following theorem.

Theorem 11.6 *The minimum number of required iterations for ϵ-convergence to the global minimum f^* for the proposed global optimization algorithm is,*

$$I^{min} = n\log_2\left[\frac{\sum_{i=1}^{n}\left(x_i^{UBD} - x_i^{LBD}\right)^2}{\sqrt{\frac{4\epsilon}{\alpha}}}\right] - 1.$$

Proof. The best case for the proposed algorithm, in terms of efficiency of partitioning, occurs when always one partition element is fathomed in every subdivision step. In this case, each non–end node in the branch and bound tree will spawn a single child node. The number of levels, NL, of this tree will be,

$$NL = n \log_2 \left[\frac{\sum_{i=1}^{n} \left(x_i^{UBD} - x_i^{LBD}\right)^2}{\sqrt{\frac{4\epsilon}{\alpha}}} \right]$$

but the total number of iterations is now,

$$I^{min} = NL - 1 = n \log_2 \left[\frac{\sum_{i=1}^{n} \left(x_i^{UBD} - x_i^{LBD}\right)^2}{\sqrt{\frac{4\epsilon}{\alpha}}} \right] - 1.$$

Remark 11.6.1 It is interesting to note that although I^{max} is exponential in the total number of variables n, which is expected since the class of global optimization problems is NP–hard, I^{min} is only linear in n. In practice, the total number of iterations is much closer to I^{min} rather than to I^{max}.

Summary

In this chapter, we discussed the global optimization algorithm αBB for box constrained nonlinear optimization problems that are twice continuously differentiable. A novel convex lower bounding function, $\mathcal{L}$, of the function f was defined, and several important theoretical properties were presented. These properties of the function $\mathcal{L}$, coupled with an efficient partitioning scheme, enabled us to construct a global optimization algorithm which was proven to always converge to a global minimum through the solution of a series of convex nonlinear optimization problems. Lower and upper bounds on the total number of required iterations were also derived.

12 THE αBB APPROACH FOR GENERAL CONSTRAINED TWICE-DIFFERENTIABLE NLPS : THEORY

In this chapter, we present the theory of the αBB global optimization approach for general constrained nonlinear optimization problems that are twice differentiable. Section 12.1 presents the formulation and an outline of the basic principles behind the αBB approach. Section 12.2 discusses (i) the underestimation strategies employed for special structure terms such as bilinear, trilinear, fractional, fractional trilinear and univariate concave terms, (ii) the novel convex underestimation functions for general nonconvex terms, as well as (iii) key theoretical results that provide the relation between the convex envelope and the maximum separation distance of bilinear terms, and the convex envelope and the maximum separation distance of the novel underestimators for general structure terms. Section 12.3 presents the overall convex underestimating functions, discusses the treatment of the equality constraints, and provides the convex lower bounding formulation. Section 12.4 focuses on novel and rigorous methods for calculating values of the α parameters that ensure convergence to the global minimum. Section 12.5 presents an example that illustrates the performance of different methods for the rigorous calculation of the α parameters. Section 12.6 presents the algorithmic steps of the αBB global optimization approach for general constrained twice differentiable NLPs. Finally, section 12.7 discusses the geometrical interpretation of the αBB approach through a highly nonlinear example. The material presented in this chapter is based on the work of Androulakis *et al.* (1995), Adjiman and Floudas (1996), and Adjiman *et al.* (1998a).

C. A. Floudas, *Deterministic Global Optimization*
© Springer Science+Business Media Dordrecht 2000

12.1 FORMULATION AND PRINCIPLES OF THE αBB GLOBAL OPTIMIZATION ALGORITHM

The αBB algorithm is based on a difference of convex functions transformation and operates within a branch-and-bound framework. It addresses nonconvex minimization problems with the following form.

$$\begin{aligned} \min_{\boldsymbol{x}} \quad & f(\boldsymbol{x}) \\ s.t. \quad & \boldsymbol{g}(\boldsymbol{x}) \leq \boldsymbol{0} \\ & \boldsymbol{h}(\boldsymbol{x}) = \boldsymbol{0} \\ & \boldsymbol{x} \in X \subseteq \Re^n \end{aligned} \tag{12.1}$$

where f, $\boldsymbol{g}$ and $\boldsymbol{h}$ belong to $\mathcal{C}^2$, the set of continuous twice-differentiable functions, and $\boldsymbol{x}$ is a vector of size n. The theoretical properties of the algorithm guarantee that such a problem can be solved to global optimality with finite ϵ-convergence.

At each iteration of the αBB algorithm an upper and a lower bound are generated while a *branching* step and a *bounding* step take place. In the bounding step, a lower bound is obtained by constructing valid convex underestimators for the functions in the problem and solving the resulting convex NLP to global optimality. An upper bound is calculated either by solving the original problem locally over each subdomain of the solution space or by performing a problem evaluation at the solution of the lower bounding problem. The identification of the global optimal solution relies upon the validity of the lower bounding problems, as well as the construction of increasingly tight lower bounding problems for successive partitions of the solution space. Such properties lead to the generation of a *nondecreasing sequence* of lower bounds which progresses towards the global optimal solution. The convergence characteristics of the αBB algorithm depend on the quality of the underestimators used. The derivation of the lower bounding problem from the original problem plays an important role in the algorithmic procedure.

12.2 UNDERESTIMATION STRATEGIES

A crucial step in generating valid underestimating functions is the decomposition of each nonlinear function into a sum of terms belonging to one of several categories: linear, bilinear, trilinear, fractional, fractional trilinear, convex, univariate concave or general nonconvex. These terms can be readily identified, and techniques can be devised in order to generate valid and in some cases very tight convex or linear underestimators. Note also that the approaches presented in Chapters 8 and 9 can be used to construct customized underestimators for other mathematical structures such as signomial expressions based on the work of Maranas and Floudas (1997). It should be noted that the linear

and convex terms do not require any transformation. The convex envelopes of the bilinear, trilinear, fractional, fractional trilinear and univariate concave terms are discussed next.

12.2.1 Underestimating Bilinear Terms

For a bilinear term, xy, McCormick (1976) and Al-Khayyal and Falk (1983) showed that the tightest convex lower bound over the domain $[x^L, x^U] \times [y^L, y^U]$ is obtained by introducing a new variable w_B such that :

$$w_B = \max\{x^L y + y^L x - x^L y^L;\ x^U y + y^U x - x^U y^U\}. \tag{12.2}$$

This lower bound can be relaxed and included in the minimization problem by adding two linear inequality constraints,

$$\begin{aligned} w_B &\geq x^L y + y^L x - x^L y^L, \\ w_B &\geq x^U y + y^U x - x^U y^U. \end{aligned} \tag{12.3}$$

Moreover, an upper bound can be imposed on w_B to better approximate the original problem (McCormick, 1976). This is achieved through the addition of two linear constraints:

$$\begin{aligned} w_B &\leq x^U y + y^L x - x^U y^L, \\ w_B &\leq x^L y + y^U x - x^L y^U. \end{aligned} \tag{12.4}$$

These two additional constraints are of particular use when the bounds of the domains are updated at each iteration.

The convex envelope for a bilinear term xy with $x \in [x^L, x^U]$ and $y \in [y^L, y^U]$ is given by Equation (12.2). This discontinuous function is piecewise linear and the following theorem (see also section 2.4.4 of Chapter 2) defines the convex envelope of a bilinear term.

Theorem 12.2.1 *The convex envelope of a bilinear term xy with $x \in [x^L, x^U]$ and $y \in [y^L, y^U]$ is*

$$w_B = \begin{cases} x^L y + y^L x - x^L y^L & \text{if } y \leq -\frac{y^U - y^L}{x^U - x^L} x + \frac{x^U y^U - x^L y^L}{x^U - x^L} \\ x^U y + y^U x - x^U y^U & \text{otherwise} \end{cases}$$

Proof. The proof and geometrical interpretation can be found in Chapter 2.

As we discussed in section 2.4.4 of Chapter 2, Androulakis *et al.* (1995) determined the maximum separation distance of a bilinear term and its convex envelope based on the following property :

Property 12.2.1 *The maximum separation of the bilinear term xy from its convex envelope,* $\max\left(x^L y + x y^L - x^L y^L,\ x^U y + x y^U - x^U y^U\right)$, *inside the rectangle* $\left[x^L, x^U\right] \times \left[y^L, y^U\right]$ *occurs at the middle point*

$$x^m = \frac{x^L + x^U}{2}, \quad y^m = \frac{y^L + y^U}{2},$$

and is equal to one fourth of the area of the rectangular domain,

$$\frac{\left(x^U - x^L\right)\left(y^U - y^L\right)}{4}.$$

Proof. The proof can be found in Chapter 2.

12.2.2 Underestimating Trilinear Terms

A trilinear term of the form xyz can be underestimated according to the approach proposed by Maranas and Floudas (1995). A new variable w_T is introduced and bounded from below by the following eight inequality constraints:

$$\begin{array}{lcl}
w_T & \geq & xy^Lz^L + x^Lyz^L + x^Ly^Lz - 2x^Ly^Lz^L, \\
w_T & \geq & xy^Uz^U + x^Uyz^L + x^Uy^Lz - x^Uy^Lz^L - x^Uy^Uz^U, \\
w_T & \geq & xy^Lz^L + x^Lyz^U + x^Ly^Uz - x^Ly^Uz^U - x^Ly^Lz^L, \\
w_T & \geq & xy^Uz^L + x^Uyz^U + x^Ly^Uz - x^Ly^Uz^L - x^Uy^Uz^U, \\
w_T & \geq & xy^Lz^U + x^Lyz^L + x^Uy^Lz - x^Uy^Lz^U - x^Ly^Lz^L, \\
w_T & \geq & xy^Lz^U + x^Lyz^U + x^Uy^Uz - x^Ly^Lz^U - x^Uy^Uz^U, \\
w_T & \geq & xy^Uz^L + x^Uyz^L + x^Ly^Lz - x^Uy^Uz^L - x^Ly^Lz^L, \\
w_T & \geq & xy^Uz^U + x^Uyz^U + x^Uy^Uz - 2x^Uy^Uz^U.
\end{array} \tag{12.5}$$

12.2.3 Underestimating Fractional Terms

Fractional terms of the form x/y are underestimated by introducing a new variable w_F and two new constraints (Maranas and Floudas, 1995) which depend on the sign of the bounds on x.

$$\begin{array}{lcl}
w_F & \geq & \begin{cases} x^L/y + x/y^U - x^L/y^U & \text{if } x^L \geq 0 \\ x/y^U - x^Ly/y^Ly^U + x^L/y^L & \text{if } x^L < 0 \end{cases} \\
w_F & \geq & \begin{cases} x^U/y + x/y^L - x^U/y^L & \text{if } x^U \geq 0 \\ x/y^L - x^Uy/y^Ly^U + x^U/y^U & \text{if } x^U < 0 \end{cases}
\end{array} \tag{12.6}$$

Adjiman *et al.* (1998d) determined the maximum separation distance of a fractional term. This basic result is represented by the following property :

Property 12.2.2 *For a fractional term x_1/x_2, the maximum separation distance expression depends on the signs of the variable bounds. For $x_2^L > 0$, the maximum separation distance is given by*

$$d_{max} = \begin{cases} \frac{x_1^L + x_1^U}{4}\left(\frac{1}{x_2^U} + \frac{1}{x_2^L}\right) & \textit{for } x_1^L \geq 0, \\ \frac{x_1^L - x_1^U}{x_2^L - x_2^U} + \frac{x_1^L\, x_2^{U\,2} - x_1^U\, x_2^{L\,2}}{x_2^L\, x_2^U\left(x_2^L - x_2^U\right)} + \frac{2\left(x_1^L - 2\,x_1^U\right)\sqrt{x_2^L}}{\left(x_2^L - x_2^U\right)\sqrt{x_2^U}} + \frac{2\,x_1^L}{\sqrt{x_2^L\, x_2^U}} & \textit{for } x_1^U < 0, \\ \max\{d_1, d_2\} & \textit{for } x_1^L < 0, \\ & x_1^U \geq 0. \end{cases} \tag{12.7}$$

where

$$d_1 = \begin{cases} -\frac{x_1^L}{x_2^L} & \text{if } x_1^L \leq \frac{x_1^L x_2^L}{x_2^L - x_2^U}, \\ 0 & \text{otherwise.} \end{cases}$$

and, given $A = 54 x_1^L x_1^U x_2^L x_2^U \left({x_1^L}^2 + \sqrt{x_1^{L^4} + x_1^L x_1^U x_2^L x_2^U} \right)$,

$$d_2 = \begin{cases} \frac{54^{\frac{2}{3}} {x_1^L}^2 x_1^U x_2^L x_2^U A^{\frac{2}{3}}}{\left(x_2^U - x_2^L\right)\left(A^{\frac{2}{3}} - 54^{\frac{2}{3}} x_1^L x_1^U x_2^L x_2^U\right)^2} - \frac{1458^{\frac{1}{3}} x_1^L x_1^U x_2^U}{\left(x_2^U - x_2^L\right)\left(A^{\frac{2}{3}} - 54^{\frac{2}{3}} x_1^L x_1^U x_2^L x_2^U\right)} \\ \quad + \frac{A^{\frac{2}{3}} - 54^{\frac{2}{3}} x_1^L x_1^U x_2^L x_2^U}{54^{\frac{1}{3}}\left(x_2^U - x_2^L\right) x_2^L A^{\frac{1}{3}}} + \frac{2x_1^U x_2^U - x_1^U x_2^L + x_1^L x_2^U - x_1^L x_2^L}{\left(x_2^U - x_2^L\right) x_2^L}, \\ \quad \text{if } {x_1^L}^4 + x_1^L x_1^U x_2^L x_2^U \geq 0, \text{ and } \frac{A^{\frac{2}{3}} - 54^{\frac{2}{3}} x_1^L x_1^U x_2^L x_2^U}{54^{\frac{1}{3}} x_1^L A^{\frac{1}{3}}} \in [x_2^L, x_2^U], \\ 0, \text{ otherwise} \end{cases}$$

Proof. The proof of this property is presented in Adjiman *et al.* (1998d).

12.2.4 Underestimating Fractional Trilinear Terms

For fractional trilinear terms, eight new constraints are required (Maranas and Floudas, 1995). The fractional trilinear term xy/z is replaced by the variable w_{FT} and the constraints for $x^L, y^L, z^L \geq 0$ are given by

$$\begin{aligned} w_{FT} &\geq xy^L/z^U + x^L y/z^U + x^L y^L/z - 2x^L y^L/z^U, \\ w_{FT} &\geq xy^L/z^U + x^L y/z^L + x^L y^U/z - x^L y^U/z^L - x^L y^L/z^U, \\ w_{FT} &\geq xy^U/z^L + x^U y/z^U + x^U y^L/z - x^U y^L/z^U - x^U y^U/z^L, \\ w_{FT} &\geq xy^U/z^U + x^U y/z^L + x^L y^U/z - x^L y^U/z^U - x^U y^U/z^L, \\ w_{FT} &\geq xy^L/z^U + x^L y/z^L + x^U y^L/z - x^U y^L/z^L - x^L y^L/z^U, \\ w_{FT} &\geq xy^U/z^U + x^U y/z^L + x^L y/z - x^L y^U/z^U - x^U y^U/z^L, \\ w_{FT} &\geq xy^L/z^U + x^L y/z^L + x^U y^L/z - x^U y^L/z^L - x^L y^L/z^U, \\ w_{FT} &\geq xy^U/z^L + x^U y/z^L + x^U y^U/z - 2x^U y^U/z^L. \end{aligned} \tag{12.8}$$

Remark 12.2.1 Notice that the aforementioned underestimators for bilinear, trilinear, fractional, and fractional trilinear terms require, for each term, the addition of a new variable and several linear or convex constraints. As a result, the size of the lower bounding problem can increase significantly, especially when there exist many such terms that need to be underestimated.

12.2.5 Underestimating Univariate Concave Terms

A univariate concave function $ut(x)$ over $[x^L, x^U]$ is easily underestimated by its convex envelope which takes the form (see section 2.4.3 of Chapter 2) :

$$ut(x^L) + \frac{ut(x^U) - ut(x^L)}{x^U - x^L}(x - x^L). \tag{12.9}$$

The generation of the best convex underestimator for a univariate concave function does not require the introduction of additional variables or constraints.

12.2.6 Underestimating General Nonconvex Terms

For the most general nonconvexities, a slightly modified version of the underestimator proposed by Maranas and Floudas (1994b) is used. A function $f(\boldsymbol{x}) \in \mathcal{C}^2(\Re^n)$ is underestimated over the entire domain $[\boldsymbol{x}^L, \boldsymbol{x}^U]$ by the function $\mathcal{L}(\boldsymbol{x})$ defined as

$$\mathcal{L}(\boldsymbol{x}) = f(\boldsymbol{x}) + \sum_{i=1}^{n} \alpha_i (x_i^L - x_i)(x_i^U - x_i) \tag{12.10}$$

where the α_i's are positive scalars defined such that the underestimating function $\mathcal{L}$ is convex.

Remark 12.2.2 Notice that in (12.10) we have each parameter α_i being associated with each variable x_i, as proposed by Androulakis *et al.* (1995), who extended the properties of the αBB from box constrained nonlinear optimization problems to general constrained nonlinear twice differentiable optimization problems. Androulakis *et al.* (1995) proved the analogous properties for general constrained problems (see properties 11.2.2-11.2.6 of Chapter 11) that form the basis for the αBB global optimization approach. These properties and their proofs follow exactly the arguments of the box constrained nonlinear problems and will not be repeated here. The reader is directed to the paper by Androulakis *et al.* (1995).

Remark 12.2.3 An important observation is that the underestimators of (12.10) do not introduce any new variables or constraints. Hence, the size of the lower bounding problem remains the same as the original regardless of the number of terms that may exist. Also, notice that the underestimators of (12.10) can be employed for special structure terms as well.

Since the summation term in Equation (12.10) is negative over the entire region $[\boldsymbol{x}^L, \boldsymbol{x}^U]$, $\mathcal{L}(\boldsymbol{x})$ is a guaranteed underestimator of $f(\boldsymbol{x})$. Furthermore, since the quadratic term is convex, all nonconvexities in the original function $f(\boldsymbol{x})$ can be overpowered given sufficiently large values of the α_i parameters: $\mathcal{L}(\boldsymbol{x})$ is therefore a *valid convex underestimator*. Since $\mathcal{L}(\boldsymbol{x})$ is convex if and only if its Hessian matrix $H_{\mathcal{L}}(\boldsymbol{x})$ is positive semi-definite, a useful convexity condition is derived by noting that $H_{\mathcal{L}}(\boldsymbol{x})$ is related to the Hessian matrix $H_f(\boldsymbol{x})$ of $f(\boldsymbol{x})$ by

$$H_{\mathcal{L}}(\boldsymbol{x}) = H_f(\boldsymbol{x}) + 2\,\Delta, \tag{12.11}$$

where Δ is a diagonal matrix whose diagonal elements are the α_i's. Δ is referred to as the *diagonal shift matrix*, since the addition of the quadratic term to the function $f(\boldsymbol{x})$, as shown in Equation (12.10), corresponds to the introduction of a *shift* in the diagonal elements of its Hessian matrix $H_f(\boldsymbol{x})$.

The following theorem can then be used to ensure that $\mathcal{L}(\boldsymbol{x})$ is indeed a convex underestimator:

Theorem 12.2.2 *$\mathcal{L}(\boldsymbol{x})$, as defined in Equation (12.10), is convex if and only if $H_f(\boldsymbol{x})+2\,\Delta = H_f(\boldsymbol{x})+2\,\mathrm{diag}\,(\alpha_i)$ is positive semi-definite for all $\boldsymbol{x} \in [\boldsymbol{x}^L, \boldsymbol{x}^U]$.*

A number of deterministic methods have been devised in order to automatically identify an appropriate diagonal shift matrix (Adjiman *et al.*, 1998a). They are discussed in detail in Section 12.4.

12.2.7 αBB Underestimators for Bilinear Terms vs Convex Envelopes

The relationship between the general underestimators of the αBB algorithm and the convex envelope for bilinear terms used by McCormick (1976) and Al-Khayyal and Falk (1983) is explored here.

Discontinuous α Parameters. Because the convex envelope for bilinear terms is a discontinuous function, no continuous function can define an underestimator of comparable quality. To generate a tight lower bound, the α values used to construct an underestimator of the form given in Equation (12.10) must be discontinuous. For region (1) of Figure 2.1, the following expressions are suggested:

$$\alpha_x = \frac{1}{2}\,\frac{y - y^L}{x^U - x}; \quad \alpha_y = \frac{1}{2}\,\frac{x - x^L}{y^U - y},$$

while for region (2) of Figure 2.1, the following can be used:

$$\alpha_x = \frac{1}{2}\,\frac{y^U - y}{x - x^L}; \quad \alpha_y = \frac{1}{2}\,\frac{x - x^L}{y^U - y}.$$

The values of α_x and α_y are non-negative for all $x \in [x^L, x^U]$ and all $y \in [y^L, y^U]$. Using the proposed α values therefore generates a *valid* underestimator throughout the domain of interest. Although the denominators of the α expressions in region (1) vanish along two borders of the rectangle ($x = x^U$ and $y = y^U$) it can be seen from Figure 2.1 that this does not pose any problem as the region (2) expressions can be used along these lines in order to evaluate α_x and α_y. Similar observations hold for the region (2) expressions.

The underestimator

$$\mathcal{L}(x, y) = xy + \alpha_x(x - x^U)(x - x^L) + \alpha_y(y - y^U)(y - y^L)$$

then becomes $x^L y + y^L x - x^L y^L$ in region (1) and $x^U y + y^U x - x^U y^U$ in region (2). It is therefore equivalent to the convex envelope of a bilinear term as defined in Theorem 12.2.1.

12.3 OVERALL VALID CONVEX UNDERESTIMATOR

Based on the underestimators discussed for each of the term types identified, a convex underestimator for any given twice-differentiable function can now be obtained through a decomposition approach. A function $f(\boldsymbol{x})$ with continuous second-order derivatives can be written as

$$\begin{aligned} f(\boldsymbol{x}) &= LT(\boldsymbol{x}) + CT(\boldsymbol{x}) + \sum_{i=1}^{bt} b_i x_{B_i,1} x_{B_i,2} + \sum_{i=1}^{tt} t_i x_{T_i,1} x_{T_i,2} x_{T_i,3} \\ &+ \sum_{i=1}^{ft} f_i \frac{x_{F_i,1}}{x_{F_i,2}} + \sum_{i=1}^{ftt} ft_i \frac{x_{FT_i,1} x_{FT_i,2}}{x_{FT_i,3}} + \sum_{i=1}^{ut} UT_i(x^i) + \sum_{i=1}^{nt} NT_i(\boldsymbol{x}) \end{aligned}$$

where $LT(\boldsymbol{x})$ is a linear term; $CT(\boldsymbol{x})$ is a convex term; bt is the number of bilinear terms, $x_{B_i,1}$ and $x_{B_i,2}$ denote the two variables that participate in the ith bilinear term and b_i is its coefficient; tt is the number of trilinear terms, $x_{T_i,1}$, $x_{T_i,2}$ and $x_{T_i,3}$ denote the three variables that participate in the ith trilinear term and t_i is its coefficient; ft is the number of fractional terms, $x_{F_i,1}$ and $x_{F_i,2}$ denote the two variables that participate in the ith fractional term and f_i is its coefficient; ftt is the number of fractional trilinear terms, $x_{FT_i,1}$, $x_{FT_i,2}$ and $x_{FT_i,3}$ denote the three variables that participate in the ith fractional trilinear term and ft_i is its coefficient; ut is the number of univariate concave terms, $UT_i(x^i)$ is the ith univariate concave term, x^i denotes the variable that participates in UT_i; nt is the number of general nonconvex terms, $NT_i(\boldsymbol{x})$ is the ith general nonconvex term.
The corresponding lower bounding function is

$$\begin{aligned} \mathcal{L}(\boldsymbol{x}, \boldsymbol{w}) = \; & LT(\boldsymbol{x}) + CT(\boldsymbol{x}) + \sum_{i=1}^{bt} b_i w_{B_i} \\ & + \sum_{i=1}^{tt} t_i w_{T_i} + \sum_{i=1}^{ft} f_i w_{F_i} + \sum_{i=1}^{ftt} ft_i w_{FT_i} \\ & + \sum_{i=1}^{ut} \left(UT_i(x^{i,L}) + \frac{UT_i(x^{i,U}) - UT_i(x^{i,L})}{x^{i,U} - x^{i,L}} (x - x^{i,L}) \right) \\ & + \sum_{i=1}^{nt} \left(NT_i(\boldsymbol{x}) + \sum_{j=1}^{n} \alpha_{ij} (x_j^L - x_j)(x_j^U - x_j) \right) \end{aligned} \tag{12.12}$$

where α_{ij} corresponds to term i and variable j and satisfies Theorem 12.2.2. The vector $\boldsymbol{w}$ includes the w_{B_i}, w_{T_i}, w_{F_i} and w_{FT_i} variables. The w_{B_i} variables are defined by Equations (12.3) and (12.4). The w_{T_i}, w_{F_i} and w_{FT_i} variables must satisfy constraints of the forms given by Equations (12.5), (12.6) and (12.8) respectively.

Remark 12.3.1 In theory, any twice-differentiable function can be treated as a single general nonconvex term. However, the decomposition of functions into terms can improve the performance of αBB algorithm in two ways. First, the use of tight underestimators for certain types of terms increases the quality of

the lower bounds generated for the optimum solution. Second, it will be shown in Section 12.4 that the construction of a convex underestimator for a general nonconvex term is dependent on the dimensionality of the term to order $\mathcal{O}(n^2)$ or higher. The separation of large terms into terms involving a smaller number of variables therefore results in a decrease in computational expense. It is worth mentioning that the structure of many physical problems lends itself naturally to such a decomposition.

12.3.1 Equality Constraints

Two strategies can be used to underestimate a nonlinear equality depending on the type of terms it involves. The first approach is used for equalities in which only linear, bilinear, trilinear, fractional and fractional trilinear terms appear. The nonlinear terms are replaced by new variables which participate linearly in the problem. The equality resulting from the substitution is therefore linear. Moreover, since the set of values these new variables can take on is a superset of the values that can be attained by the nonlinear terms, the linear equality corresponds to an enlarged feasible region. Thus, given the equality

$$\begin{aligned} LT(\boldsymbol{x}) + \sum_{i=1}^{bt} b_i x_{B_i,1} x_{B_i,2} + \sum_{i=1}^{tt} t_i x_{T_i,1} x_{T_i,2} x_{T_i,3} \\ + \sum_{i=1}^{ft} f_i \frac{x_{F_i,1}}{x_{F_i,2}} + \sum_{i=1}^{ftt} ft_i \frac{x_{FT_i,1} x_{FT_i,2}}{x_{FT_i,3}} = 0, \end{aligned}$$

the following underestimator can be used

$$LT(\boldsymbol{x}) + \sum_{i=1}^{bt} b_i w_{B_i} + \sum_{i=1}^{tt} t_i w_{T_i} + \sum_{i=1}^{ft} f_i w_{F_i} + \sum_{i=1}^{ftt} ft_i w_{FT_i} = 0,$$

where the notation is as previously specified, and the appropriate inequality constraints for the $\boldsymbol{w}$ variables are added to the problem.

The second approach is used when the nonlinear equality constraints involve convex, univariate concave, or general nonconvex terms. If the nonlinear equality constraint contains convex or general nonconvex terms, the equality obtained by simple substitution of the corresponding underestimators is nonlinear. If it contains univariate concave terms, it is linear but it corresponds to a different feasible region. In the presence of convex, general nonconvex or univariate concave terms, the original equality $h(\boldsymbol{x}) = 0$ must therefore be rewritten as two inequalities of opposite signs,

$$\begin{cases} h(\boldsymbol{x}) & \leq 0 \\ -h(\boldsymbol{x}) & \leq 0. \end{cases}$$

These two inequalities must then be underestimated independently. The univariate concave terms appearing in the nonconvex equality become convex in

one of the two inequalities while the convex terms become concave and the general nonconvex terms remain nonconvex.

12.3.2 Convex Lower Bounding Problem Formulation

Based on the underestimating strategies introduced earlier, a convex relaxation for any nonconvex problem of type (12.1) that belongs to the class of twice continuously differentiable continuous NLPs can be constructed as :

$$\begin{array}{ll} \min\limits_{\boldsymbol{x},\boldsymbol{w}} & \breve{f}(\boldsymbol{x},\boldsymbol{w}) \\ s.t. & \breve{\boldsymbol{g}}(\boldsymbol{x},\boldsymbol{w}) \leq \boldsymbol{0} \\ & \overset{\smile}{\boldsymbol{h}_L}(\boldsymbol{x},\boldsymbol{w}) = \boldsymbol{0} \\ & \overset{\smile}{\boldsymbol{h}_N^+}(\boldsymbol{x},\boldsymbol{w}) \leq \boldsymbol{0} \\ & \boldsymbol{h}_N^-(\boldsymbol{x},\boldsymbol{w}) \leq \boldsymbol{0} \\ & \mathcal{C}(\boldsymbol{x},\boldsymbol{w}_B) \\ & \boldsymbol{x} \in [\boldsymbol{x}^L,\boldsymbol{x}^U], \boldsymbol{w} \in [\boldsymbol{w}^L,\boldsymbol{w}^U] \end{array} \tag{12.13}$$

where the $\breve{}$ superscript denotes the convex underestimator of the specified function over the domain $\boldsymbol{x} \in [\boldsymbol{x}^L,\boldsymbol{x}^U]$; $\boldsymbol{h}_L(\boldsymbol{x})$ is the vector of equality constraints which involve only linear, bilinear, trilinear, fractional and fractional trilinear terms; the other equality constraints are represented by the vector $\boldsymbol{h}_N(\boldsymbol{x})$; $\boldsymbol{w}$ denotes the set of variables that replace bilinear, trilinear, fractional and fractional trilinear terms; and $\mathcal{C}(\boldsymbol{x},\boldsymbol{w})$ denotes the set of additional constraints that arise from the underestimation of these terms.

To develop the rigorous formulation of a valid convex lower bounding problem, it remains to select appropriate values for the α parameters in Equation (12.10).

12.4 RIGOROUS CALCULATION OF α FOR GENERAL NLPS

The focus of this section is the development of methods that rigorously generate an appropriate diagonal shift matrix, that is, a set of α parameters satisfying Theorem 12.2.2. This allows the construction of a convex underestimator $\mathcal{L}(\boldsymbol{x})$ for a twice-differentiable function $f(\boldsymbol{x})$ over a specified domain. Two classes of approaches to this problem are defined:

- Uniform diagonal shift of the Hessian matrix of $f(\boldsymbol{x})$,
- Non-uniform diagonal shift of the Hessian matrix of $f(\boldsymbol{x})$.

Prior to the presentation of the rigorous approaches for the calculation of α, two fundamental issues must be examined: the computational complexity of the α calculation problem and the design of criteria for the assessment of the calculation methods.

12.4.1 Tractability of α Calculation Techniques

As seen in Equation (12.11) and Theorem 12.2.2, the diagonal shift matrix Δ is closely linked to the Hessian matrix $H_f(\boldsymbol{x})$ of the function being underestimated. For general twice-differentiable functions, the elements of the Hessian matrix $H_f(\boldsymbol{x})$ are likely to be nonlinear functions of the variables, so that the derivation of a matrix Δ valid over the entire underestimation domain is a very difficult task. Satisfying the convexity condition of Theorem 12.2.2 is essential for the preservation of the guarantee of global optimality. The difficulties arising from the presence of the variables in the convexity condition can be alleviated through the transformation of the exact $\boldsymbol{x}$-dependent Hessian matrix to an interval matrix $[H_f]$ such that $H_f(\boldsymbol{x}) \subseteq [H_f]$, $\forall \boldsymbol{x} \in [\boldsymbol{x}^L, \boldsymbol{x}^U]$. The elements of the original Hessian matrix are treated as independent when calculating their natural interval extensions (Ratschek and Rokne, 1984; Neumaier, 1990). The interval Hessian matrix family $[H_f]$ is then used to formulate a theorem in which the α calculation problem is relaxed.

Theorem 12.4.1 *Consider a general function $f(\boldsymbol{x})$ with continuous second-order derivatives and its Hessian matrix $H_f(\boldsymbol{x})$. Let $\mathcal{L}(\boldsymbol{x})$ be defined by Equation (12.10). Let $[H_f]$ be a real symmetric interval matrix such that $H_f(\boldsymbol{x}) \subseteq [H_f]$, $\forall \boldsymbol{x} \in [\boldsymbol{x}^L, \boldsymbol{x}^U]$. If the matrix $[H_{\mathcal{L}}]$ defined by $[H_{\mathcal{L}}] = [H_f] + 2\ \Delta = [H_f] + 2\ \mathrm{diag}\,(\alpha_i)$ is positive semi-definite, then $\mathcal{L}(\boldsymbol{x})$ is convex over the domain $[\boldsymbol{x}^L, \boldsymbol{x}^U]$.*

Remark 12.4.1 The domain of validity of the underestimator, $[\boldsymbol{x}^L, \boldsymbol{x}^U]$, participates in the interval convexity condition implicitly, through the interval matrix. The use of interval arithmetic serves two purposes: it reduces the computational complexity of the α calculations and it allows the preservation and transfer of global information. These advantages come however with the penalty of obtaining values of the α parameters that are not necessarily the tightest possible.

12.4.2 Assessment of α Calculation Methods

One metric of the quality of the underestimator generated by any given α calculation method is the separation distance between the nonconvex function and its underestimator. The tighter the lower bounding scheme is, the faster the convergence becomes. For this purpose, the maximum separation distance between $f(\boldsymbol{x})$ and $\mathcal{L}(\boldsymbol{x})$, d_{max}, can be used. As we discussed in Chapter 11, Maranas and Floudas (1994b), and Androulakis *et al.* (1995) showed that it is directly proportional to the α_i's and given by

$$d_{max} = \max_{\boldsymbol{x}^L \leq \boldsymbol{x} \leq \boldsymbol{x}^U} (f(\boldsymbol{x}) - \mathcal{L}(\boldsymbol{x})) = \frac{1}{4} \sum_{i=1}^{n} \alpha_i (\boldsymbol{x}_i^U - \boldsymbol{x}_i^L)^2 \tag{12.14}$$

The complexity analysis presented in Chapter 11, showed that the α parameters and the bounds on the variables affect the maximum number of iterations required in order to achieve ϵ-convergence.

12.4.3 Uniform Diagonal Shift Matrix

For this class of methods, the underestimator $\mathcal{L}(\boldsymbol{x})$ is re-formulated using a single α value:

$$\mathcal{L}(\boldsymbol{x}) = f(\boldsymbol{x}) + \alpha \sum_i (x_i^L - x_i)(x_i^U - x_i). \tag{12.15}$$

All the nonzero elements of the diagonal shift matrix Δ are therefore equal to α. As we discussed in Chapter 11, Maranas and Floudas (1994b) showed that $\mathcal{L}(\boldsymbol{x})$, as defined by Equation (12.15), is convex if and only if

$$\alpha \geq \max\{0, -\frac{1}{2} \min_{i, \boldsymbol{x}^L \leq \boldsymbol{x} \leq \boldsymbol{x}^U} \lambda_i(\boldsymbol{x})\} \tag{12.16}$$

where the $\lambda_i(\boldsymbol{x})$'s are the eigenvalues of $H_f(\boldsymbol{x})$, the Hessian matrix of the function $f(\boldsymbol{x})$.

If $f(\boldsymbol{x})$ is convex, all eigenvalues of $H_f(\boldsymbol{x})$ are nonnegative for any $\boldsymbol{x} \in [\boldsymbol{x}^L, \boldsymbol{x}^U]$ and, by Equation (12.16), $\alpha = 0$: the original function appears unchanged in the lower bounding problem. For a nonconvex function, a measure of the degree of nonconvexity of the function is introduced through the use of the most negative eigenvalue in the construction of the underestimator: the more nonconvex the function, the smaller its minimum eigenvalue and hence the larger α becomes.

A valid lower bound on the minimum eigenvalue of $H_f(\boldsymbol{x})$ can be more easily obtained when the interval Hessian matrix $[H_f] \supseteq H_f(\boldsymbol{x})$ is introduced, as suggested by Adjiman and Floudas (1996) and Adjiman *et al.* (1998a). All the methods presented in this section generate a single α value which satisfies the following sufficient condition for the convexity of $\mathcal{L}(\boldsymbol{x})$:

$$\alpha \geq \left\{0, -\frac{1}{2}\lambda_{min}\left([H_f]\right)\right\} \tag{12.17}$$

where $\lambda_{min}\left([H_f]\right)$ is the minimum eigenvalue of the interval matrix family $[H_f]$.

One $\mathcal{O}(n^2)$ method and a variety of $\mathcal{O}(n^3)$ methods have been developed to compute a bound on the minimum eigenvalue of a symmetric interval matrix.

12.4.3.1 $\mathcal{O}(n^2)$ Method.

Method I.1: Gerschgorin's Theorem for Interval Matrices This first method is the straightforward extension of Gerschgorin's theorem (Gerschgorin,

1931) to interval matrices. Its computational complexity is only of order n^2, but the bounds it provides on the eigenvalues are often loose.

For a real matrix $A = (a_{ij})$, the well-known theorem states that the eigenvalues are bounded below by λ_{min} such that

$$\lambda_{min} = \min_i \left(a_{ii} - \sum_{j \neq i} |a_{ij}| \right). \tag{12.18}$$

In the case of interval matrices, the following extended theorem can be used.

Theorem 12.4.2 *For an interval matrix* $[A] = ([\underline{a}_{ij}, \overline{a}_{ij}])$, *a lower bound on the minimum eigenvalue is given by*

$$\lambda_{min} \geq \min_i \left[\underline{a}_{ii} - \sum_{j \neq i} \max\left(|\underline{a}_{ij}|, |\overline{a}_{ij}|\right) \right].$$

Proof. By definition, $\lambda_{min}([A]) \geq \min_{A \in [A]} \lambda_{min}(A)$. Therefore,

$$\begin{aligned}
\lambda_{min}([A]) &\geq \min_{A \in [A]} \min_i \left(a_{ii} - \sum_{j \neq i} |a_{ij}| \right) \\
&\geq \min_i \left[\min_{A \in [A]} (a_{ii}) - \max_{A \in [A]} \left(\sum_{j \neq i} |a_{ij}| \right) \right] \\
&\geq \min_i \left[\underline{a}_{ii} - \sum_{j \neq i} \max\left(|\underline{a}_{ij}|, |\overline{a}_{ij}|\right) \right].
\end{aligned}$$

12.4.3.2 $\mathcal{O}(n^3)$ Methods. The following definitions are needed for $\mathcal{O}(n^3)$ methods. Given an interval matrix $[A]$ with elements $[\underline{a}_{ij}, \overline{a}_{ij}]$,

- its radius matrix $\Delta A = (\Delta a_{ij})$ is such that $\Delta a_{ij} = \frac{\overline{a}_{ij} - \underline{a}_{ij}}{2}$,
- its *modified* radius matrix $\widetilde{\Delta A} = (\widetilde{\Delta a}_{ij})$ is such that

$$\widetilde{\Delta a}_{ij} = \begin{cases} 0 & \text{if } i = j \\ \Delta a_{ij} & \text{otherwise} \end{cases}$$

- its midpoint matrix $A_M = (a_{M,ij})$ is such that $a_{M,ij} = \frac{\overline{a}_{ij} + \underline{a}_{ij}}{2}$,
- its *modified* midpoint matrix $\tilde{A}_M = (\tilde{a}_{M,ij})$ is such that

$$\tilde{a}_{M,ij} = \begin{cases} \underline{a}_{ij} & \text{if } i = j \\ a_{M,ij} & \text{otherwise} \end{cases}$$

- a vertex matrix $A_v = (a_{v,ij})$ of $[A]$ is such that $a_{v,ij} = \underline{a}_{ij}$ or $a_{v,ij} = \overline{a}_{ij}$.

Note that $\Delta A = \widetilde{\Delta A} + \text{diag}(\Delta A)$ and $A_M = \tilde{A}_M + \text{diag}(\Delta A)$.

Method I.2: E-Matrix Method This method is an extension of the theorems developed by Deif (1991) and Rohn (1998) for the calculation of a lower bound on all the eigenvalues of an interval matrix. While they obtained expressions for the real and imaginary parts of the matrices, only the real parts are of concern for the symmetric matrices being considered here.

The following lemma is required to derive the main result for this method.

Lemma. Let $[a] = [\underline{a}, \overline{a}]$ be a single interval with midpoint a_M and radius Δa. Then for all scalars γ and all $a \in [a]$,

$$\gamma\, a \geq \gamma a_M - |\gamma| \Delta a. \tag{12.19}$$

Proof. If $\gamma = 0$, Equation (12.19) is trivially valid. For $\gamma \neq 0$,

$$\gamma(a - a_M) \geq -|\gamma|\Delta a \Leftrightarrow \begin{cases} (a_M - a) \leq \Delta a & \text{for } \gamma > 0 \\ -(a_M - a) \leq \Delta a & \text{for } \gamma < 0 \end{cases}$$

The right-hand side is always true because the distance between the midpoint and any point in the interval can never be greater than the radius of the interval.

Theorem 12.4.3 *Let E be an arbitrary real symmetric matrix. Given a symmetric interval matrix $[A]$, its modified midpoint matrix $\tilde{A}_M$ and its modified radius matrix $\widetilde{\Delta A}$, the minimum eigenvalue $\lambda_{min}(A)$ of any symmetric matrix $A \in [A]$ is such that*

$$\lambda_{min}(A) \geq \lambda_{min}\left(\tilde{A}_M + E\right) - \rho\left(\widetilde{\Delta A} + |E|\right)$$

where $\rho(M)$ denotes the spectral radius of a matrix M, i.e. its maximal absolute eigenvalue, and $|E|$ is the absolute value taken componentwise.

Proof. For all $A = (a_{ij}) \in [A]$,

$$\begin{aligned} \boldsymbol{x}^T A \boldsymbol{x} &= \sum_i a_{ii} x_i^2 + \sum_{j \neq i} x_i a_{ij} x_j & (12.20) \\ &\geq \sum_i \underline{a}_{ii} x_i^2 + \sum_{j \neq i} x_i a_{ij} x_j. & (12.21) \end{aligned}$$

Setting $x_i x_j = \gamma$ and using the Lemma (12.19) in the second term of the right hand side of (12.21), we find

$$\boldsymbol{x}^T A \boldsymbol{x} \geq \sum_i \underline{a}_{ii} x_i^2 + \sum_{j \neq i} x_i a_{M,ij} x_j - \sum_{j \neq i} |x_i| \Delta a_{ij} |x_j|. \tag{12.22}$$

The matrix $E = (e_{ij})$ is now introduced. Since $\sum_i e_{ii} x_i^2 - \sum_i e_{ii} x_i^2 + \sum_{j \neq i} x_i e_{ij} x_j - \sum_{j \neq i} x_i e_{ij} x_j = 0$, this term can be added to the right hand side of (12.22) to yield

$$\begin{aligned} \boldsymbol{x}^T A\boldsymbol{x} \;\geq\; & \sum_i (\underline{a}_{ii} + e_{ii})x_i^2 + \sum_{j\neq i} x_i(a_{M,ij} + e_{ij})x_j \\ & - \sum_i e_{ii}x_i^2 - \sum_{j\neq i} (|x_i|\Delta a_{ij}|x_j| + x_i e_{ij} x_j)\,. \end{aligned} \tag{12.23}$$

The right hand side of Equation (12.23) can be further relaxed:

$$\begin{aligned} \boldsymbol{x}^T A\boldsymbol{x} \;\geq\; & \sum_i (\underline{a}_{ii} + e_{ii})x_i^2 + \sum_{j\neq i} x_i(a_{M,ij} + e_{ij})x_j \\ & - \sum_i (0 + |e_{ii}|)\, x_i^2 - \sum_{j\neq i} |x_i|\,(\Delta a_{ij} + |e_{ij}|)\,|x_j|. \end{aligned} \tag{12.24}$$

This is equivalent to

$$\boldsymbol{x}^T A\boldsymbol{x} \geq \boldsymbol{x}^T(\tilde{A}_M + E)\boldsymbol{x} - |\boldsymbol{x}^T|(\widetilde{\Delta A} + |E|)|\boldsymbol{x}|. \tag{12.25}$$

Using the Rayleigh quotient, we have $\lambda_{min}(A) = \min_{\boldsymbol{x}^T\boldsymbol{x}=1} \boldsymbol{x}^T A\boldsymbol{x}$. Hence,

$$\begin{aligned} \lambda_{min}(A) \;\geq\; & \min_{\boldsymbol{x}^T\boldsymbol{x}=1} \boldsymbol{x}^T(\tilde{A}_M + E)\boldsymbol{x} - \max_{\boldsymbol{x}^T\boldsymbol{x}=1} |\boldsymbol{x}^T|(\widetilde{\Delta A} + |E|)|\boldsymbol{x}| \\ \geq\; & \lambda_{min}(\tilde{A}_M + E) - \rho(\widetilde{\Delta A} + |E|). \end{aligned}$$

The optimal choice for the matrix E is a matrix that minimizes the lower bound, λ_{min}, obtained. Unfortunately, this matrix cannot be determined *a priori*. Two choices for the matrix E have been used to illustrate this α calculation method:

- $E = 0$,
- $E = \mathrm{diag}\,(\Delta A)$.

The second choice of E matrix yields a result equivalent to the theorems of Deif (1991) and Rohn (1998), namely:

$$\lambda_{min}(A) \geq \lambda_{min}\,(A_M) - \rho\,(\Delta A)\,. \tag{12.26}$$

Method I.3: Mori and Kokame's Method Mori and Kokame (1994) suggested the use of the lower and upper vertex matrices, $\underline{A} = (\underline{a}_{ij})$ and $\overline{A} = (\overline{a}_{ij})$ respectively, in order to obtain a lower bound on the minimum eigenvalue of an interval matrix.

Theorem 12.4.4 *For any symmetric matrix A in the symmetric interval matrix family $[A]$, the minimum eigenvalue $\lambda_{min}(A)$ of A is such that*

$$\lambda_{min}(A) \geq \lambda_{min}(\underline{A}) - \rho(\overline{A} - \underline{A}).$$

The Mori and Kokame method can be compared to the E-matrix method with $E = \text{diag}(\Delta A)$. Since $\overline{A} - \underline{A} = 2\Delta A$, the value provided by the Mori and Kokame method is greater than $\lambda_{min}(\underline{A}) - 2\rho(\Delta A)$. Comparing this with Equation (12.26), the E-matrix method with $E = \text{diag}(\Delta A)$ yields better results if and only if $\lambda_{min}(\underline{A}) - \lambda_{min}(A_M) \leq \rho(\Delta A)$. For any vector $\boldsymbol{x}$,

$$\lambda_{min}(\underline{A}) \leq \frac{\boldsymbol{x}^T \underline{A} \boldsymbol{x}}{\boldsymbol{x}^T \boldsymbol{x}} = \frac{\boldsymbol{x}^T A_M \boldsymbol{x}}{\boldsymbol{x}^T \boldsymbol{x}} - \frac{\boldsymbol{x}^T \Delta A \boldsymbol{x}}{\boldsymbol{x}^T \boldsymbol{x}} \leq \frac{\boldsymbol{x}^T A_M \boldsymbol{x}}{\boldsymbol{x}^T \boldsymbol{x}} + \rho(\Delta A).$$

In particular

$$\lambda_{min}(\underline{A}) \leq \lambda_{min}(A_M) + \rho(\Delta A).$$

The E-matrix method with $E = \text{diag}(\Delta A)$ is therefore at least as good as the Mori and Kokame method.

Method I.4: The Lower Bounding Hessian Method Unlike other $\mathcal{O}(n^3)$ methods, this technique requires the construction of a single real matrix, referred to as a *lower bounding Hessian matrix*, in order to determine a lower bound on the minimum eigenvalue of the interval Hessian matrix. Using a necessary and sufficient condition proved by Stephens (1997), a bounding Hessian can be defined in terms of a property of its quadratic form.

Definition 12.4.1 *Given a symmetric interval Hessian matrix* $[A]$, *the real symmetric matrix* L *is a lower bounding Hessian of* $[A]$ *if and only if* $\boldsymbol{x}^T L \boldsymbol{x} \leq \boldsymbol{x}^T A \boldsymbol{x}$, $\forall \boldsymbol{x} \in \Re^n$, $\forall A \in [A]$. *Similarly, the real symmetric matrix* U *is an upper bounding Hessian of* $[A]$ *if and only if* $\boldsymbol{x}^T U \boldsymbol{x} \geq \boldsymbol{x}^T A \boldsymbol{x}$, $\forall \boldsymbol{x} \in \Re^n$, $\forall A \in [A]$.

It follows immediately from this definition that the minimum eigenvalue of a lower bounding Hessian is a guaranteed lower bound on the minimum eigenvalue of the interval matrix family.

The procedure proposed by Stephens (1997) for the construction of upper bounding Hessians can be appropriately transformed to build a lower bounding Hessian.

Theorem 12.4.5 *Given an interval Hessian matrix* $[A] = ([\underline{a}_{ij}, \overline{a}_{ij}])$, *the matrix* $L = (l_{ij})$ *where*

$$l_{ij} = \begin{cases} \underline{a}_{ii} + \sum\limits_{k \neq i} \frac{\underline{a}_{ik} - \overline{a}_{ik}}{2} & i = j \\ \frac{\underline{a}_{ij} + \overline{a}_{ij}}{2} & i \neq j \end{cases}$$

is a lower bounding Hessian of $[A]$.

Proof. Let $[A]$ be an $n \times n$ symmetric interval matrix. Recall that a vertex matrix A_v of $[A]$ is defined by $(A_v)_{ij} = \underline{a}_{ij}$ or $(A_v)_{ij} = \overline{a}_{ij}$.

Let X be the set of n-dimensional vectors with unit norm. For any $\boldsymbol{x} \in X$, the vertex matrix $A_v^* = (a_{v,ij}^*)$ is defined as

$$a_{v,ij}^* = \begin{cases} \underline{a}_{ij} & \text{if } x_i x_j \geq 0 \\ \overline{a}_{ij} & \text{if } x_i x_j < 0 \end{cases}$$

Then, $\boldsymbol{x}^T A_v^* \boldsymbol{x} \leq \boldsymbol{x}^T A \boldsymbol{x} = \sum_{i=1}^{n} a_{ii} x_i^2 + \sum_{i=1}^{n} \sum_{j \neq i} a_{ij} x_i x_j$, $\forall \boldsymbol{x} \in X$, where $A \in [A]$ is a real symmetric matrix. Consequently, L is a lower bounding Hessian if and only if $\boldsymbol{x}^T L \boldsymbol{x} \leq \boldsymbol{x}^T A_v^* \boldsymbol{x}$, $\forall \boldsymbol{x} \in X$.

Given a vector $\boldsymbol{x} \in X$, the quadratic form for the matrix L defined in Theorem 12.4.5 is expressed as

$$\begin{aligned} \boldsymbol{x}^T L \boldsymbol{x} &= \sum_{i=1}^{n} l_{ii} x_i^2 + \sum_{i=1}^{n} \sum_{j \neq i} l_{ij} x_i x_j \\ &= \sum_{i=1}^{n} \underline{a}_{ii} x_i^2 + \sum_{i=1}^{n} \sum_{j \neq i} \tfrac{\underline{a}_{ij} - \overline{a}_{ij}}{2} x_i^2 + \sum_{i=1}^{n} \sum_{j \neq i} \tfrac{\underline{a}_{ij} + \overline{a}_{ij}}{2} x_i x_j. \end{aligned}$$

Using the fact that $[A]$ is symmetric, this is equivalent to

$$\boldsymbol{x}^T L \boldsymbol{x} = \sum_{i=1}^{n} \underline{a}_{ii} x_i^2 + 2 \sum_{i=1}^{n} \sum_{j>i} \left(\frac{\underline{a}_{ij} - \overline{a}_{ij}}{4} (x_i^2 + x_j^2) + \frac{\underline{a}_{ij} + \overline{a}_{ij}}{2} x_i x_j \right). \quad (12.27)$$

For the vertex matrix A_v^*, we have

$$\begin{aligned} \boldsymbol{x}^T A_v^* \boldsymbol{x} &= \sum_{i=1}^{n} a_{v,ii}^* x_i^2 + \sum_{i=1}^{n} \sum_{j \neq i} a_{v,ij}^* x_i x_j \\ &= \sum_{i=1}^{n} \underline{a}_{ii} x_i^2 + \sum_{i=1}^{n} \sum_{j \neq i} a_{v,ij}^* x_i x_j \\ &= \sum_{i=1}^{n} \underline{a}_{ii} x_i^2 + 2 \sum_{i=1}^{n} \sum_{j>i} a_{v,ij}^* x_i x_j. \end{aligned} \quad (12.28)$$

Comparing Equations (12.27) and (12.28), we find that $\boldsymbol{x}^T L \boldsymbol{x} \leq \boldsymbol{x}^T A_v^* \boldsymbol{x}$ if and only if

$$\sum_{i=1}^{n} \sum_{j>i} \left(\frac{\underline{a}_{ij} - \overline{a}_{ij}}{4} (x_i^2 + x_j^2) + \frac{\underline{a}_{ij} + \overline{a}_{ij}}{2} x_i x_j \right) \leq \sum_{i=1}^{n} \sum_{j>i} a_{v,ij}^* x_i x_j. \quad (12.29)$$

In order to prove that this relation holds for all $\boldsymbol{x} \in X$, we first note that, since $(x_i \pm x_j)^2 \geq 0$, $x_i^2 + x_j^2 \geq \pm 2 x_i x_j$. Hence,

$$\frac{\underline{a}_{ij} - \overline{a}_{ij}}{4} (x_i^2 + x_j^2) \leq \pm \frac{\underline{a}_{ij} - \overline{a}_{ij}}{2} x_i x_j$$

and

$$\begin{aligned}\frac{\underline{a}_{ij}-\overline{a}_{ij}}{4}(x_i^2+x_j^2)+\frac{\underline{a}_{ij}+\overline{a}_{ij}}{2}x_ix_j &\leq \pm\frac{\underline{a}_{ij}-\overline{a}_{ij}}{2}x_ix_j+\frac{\underline{a}_{ij}+\overline{a}_{ij}}{2}x_ix_j \\ &\leq \begin{cases}\underline{a}_{ij}x_ix_j \\ \overline{a}_{ij}x_ix_j\end{cases} \\ &\leq a^*_{v,ij}x_ix_j.\end{aligned}$$

Summing over all i and $j > i$, the desired relation is obtained, proving that L is a lower bounding Hessian of $[A]$.

Remark 12.4.2 Rather than calculating the minimum eigenvalue of L in order to construct an underestimator of the form given in Equation (12.10), Stephens (1997) recommends the incorporation of the lower bounding Hessian into the lower bounding function. He proves that given any twice-differentiable function $f(\boldsymbol{x})$ and its lower bounding Hessian L_f over the solution space, the function $f(\boldsymbol{x}) - 1/2\,\boldsymbol{x}^T L_f \boldsymbol{x}$ is convex. However, this expression is not less than $f(\boldsymbol{x})$ everywhere, and the addition of a constant c is necessary to ensure that a valid convex underestimator is indeed obtained. The underestimating function is then expressed as

$$f(\boldsymbol{x}) - \frac{1}{2}\boldsymbol{x}^T L_f \boldsymbol{x} + c \tag{12.30}$$

where the condition $c \leq 1/2\ \min \boldsymbol{x}^T L_f \boldsymbol{x}$ must be satisfied in order to guarantee valid underestimation throughout the solution domain. The rigorous calculation of c requires the solution of an indefinite quadratic program. The underestimator thus obtained differs from the general αBB underestimators in several ways: Equation (12.30) does not match the original function at the end points and no bound can be provided for the maximum separation distance between $f(\boldsymbol{x})$ and its underestimator. In addition, the function $f(\boldsymbol{x})$ is underestimated even when its lower bounding Hessian is positive semi-definite.

Method I.5: Hertz's Method The methods described so far either provide a lower bound on the minimum eigenvalue of the interval Hessian matrix or the smallest real part of the roots of the interval characteristic polynomial. The Hertz method allows the computation of the *exact* smallest eigenvalue of the interval Hessian matrix. This gain in accuracy comes at a cost since the number of matrices that need to be constructed in order to arrive at this value is no longer independent of the dimensionality of the problem. As this technique calls for 2^{n-1} vertex matrices, its computational cost may become prohibitive for larger problems.

The Hertz method has been described in detail in Hertz (1992) and Adjiman and Floudas (1996). A finite subset of real matrices is constructed from the interval matrix, with the property that its minimum eigenvalue is equal to the minimum eigenvalue of the interval matrix. The required real matrices can be obtained through a systematic procedure.

Theorem 12.4.6 *Let $[A] = ([\underline{a}_{ij}, \overline{a}_{ij}])$ be a symmetric interval Hessian matrix. Consider a vector $\boldsymbol{x} \in R^n$. There are 2^{n-1} possible combinations for the signs of the $x_i x_j$ products $(i \neq j)$. For the kth such combination, let the vertex matrix $A_k \in [A]$ be defined by $A_k = [a_{ij}^k]$ where*

$$a_{ij}^k = \begin{cases} \underline{a}_{ii} & \text{if } i = j, \\ \underline{a}_{ij} & \text{if } x_i x_j \geq 0, i \neq j, \\ \overline{a}_{ij} & \text{if } x_i x_j < 0, i \neq j. \end{cases}$$

Then the smallest eigenvalue of the set of matrices $\{A_k\}$ is the minimum eigenvalue of $[A]$.

12.4.4 Non-Uniform Diagonal Shift Matrix

This class of methods allows the calculation of a different α value for each variable in order to construct an underestimator of the form shown in Equation (12.10). The non-zero elements of the diagonal shift matrix Δ can no longer be related to the minimum eigenvalue of the interval Hessian matrix $[H_f]$. The condition presented in Theorem 12.4.1 for the convexity of the underestimator cannot be simplified.

Before presenting rigorous procedures for the derivation of a diagonal shift matrix Δ such that $[H_f] + 2\Delta$ is positive semi-definite, a few definitions are introduced.

Definition 1. A square matrix A is an M-**matrix** if all its off-diagonal elements are non-positive and there exists a real positive vector $\boldsymbol{u}$ such that $A\,\boldsymbol{u} > 0$. Inequalities are understood component wise.

Definition 2. The **comparison matrix** $\langle A \rangle$ of the interval matrix $[A] = ([\underline{a}_{ij}, \overline{a}_{ij}])$ is

$$\langle A \rangle_{ij} = \begin{cases} 0 & \text{if } i = j \text{ and } 0 \in [\underline{a}_{ij}, \overline{a}_{ij}] \\ \min\{|\underline{a}_{ij}|, |\overline{a}_{ij}|\} & \text{if } i = j \text{ and } 0 \notin [\underline{a}_{ij}, \overline{a}_{ij}] \\ -|a|_{ij} & \text{if } i \neq j \end{cases}$$

where $|a|_{ij} = \max\{|\underline{a}_{ij}|, |\overline{a}_{ij}|\}$.

Definition 3. A square interval matrix $[A]$ is an H-**matrix** if its comparison matrix $\langle A \rangle$ is an M-matrix. It can be shown that if $[A]$ is an H-matrix, it is regular and 0 is not an eigenvalue of $[A]$ (Neumaier, 1992).

The first rigorous method belonging to this class is a $\mathcal{O}(n^2)$ method, while the two other methods presented are iterative, with varying degrees of complexity.

12.4.4.1 $\mathcal{O}(n^2)$ Method.

Method II.1: Scaled Gerschgorin Theorem

Theorem 12.4.7 *For any vector $\boldsymbol{d} > 0$ and a symmetric interval matrix $[A]$, define the vector α as*

$$\alpha_i = \max\left\{0, -\frac{1}{2}\left(\underline{a}_{ii} - \sum_{j\neq i} |a|_{ij}\frac{d_j}{d_i}\right)\right\}$$

where $|a|_{ij} = \max\{|\underline{a}_{ij}|, |\overline{a}_{ij}|\}$.

Then, for all $A \in [A]$, the matrix $A_{\mathcal{L}} = A + 2\Delta$ with $\Delta = \mathrm{diag}(\alpha_i)$ is positive semi-definite.

Proof. First, we show that the diagonal elements of $A_{\mathcal{L}}$ are non-negative. By definition,

$$a_{\mathcal{L},ii}d_i = a_{ii}d_i + \max\left\{0, -\underline{a}_{ii}d_i + \sum_{j\neq i}|a|_{ij}d_j\right\}.$$

<u>If $a_{ii} \geq 0$:</u> Since $d_i > 0$, $a_{\mathcal{L},ii}d_i$ is the sum of two non-negative terms and therefore $a_{\mathcal{L},ii} > 0$.
<u>If $a_{ii} < 0$:</u> Since $\underline{a}_{ii} \leq a_{ii}$, $\underline{a}_{ii} < 0$. Therefore,

$$\underline{a}_{ii}d_i - \sum_{j\neq i}|a|_{ij}d_j < 0.$$

Hence,

$$a_{\mathcal{L},ii}d_i = a_{ii}d_i - \underline{a}_{ii}d_i + \sum_{j\neq i}|a|_{ij}d_j \geq 0.$$

The diagonal elements of $A_{\mathcal{L}}$ are therefore non-negative. Using this property, the comparison matrix $\langle A_{\mathcal{L}}\rangle$ of $A_{\mathcal{L}}$ is given by

$$\begin{aligned}\langle A_{\mathcal{L}}\rangle_{ij} &= \begin{cases} \underline{a}_{\mathcal{L},ii} & \text{if } i = j \\ -|a_{\mathcal{L}}|_{ij} & \text{if } i \neq j \end{cases} \\ &= \begin{cases} \underline{a}_{ii} + \max\left\{0, -\underline{a}_{ii} + \sum_{k\neq i}|a|_{ik}\frac{d_k}{d_i}\right\} & \text{if } i = j \\ -|a|_{ij} & \text{if } i \neq j \end{cases}\end{aligned}$$

We now prove that $\langle A_{\mathcal{L}}\rangle\boldsymbol{d} \geq 0$. The ith component of the vector $\langle A_{\mathcal{L}}\rangle\boldsymbol{d}$ is given by

$$(\langle A_{\mathcal{L}}\rangle\boldsymbol{d})_i = \underline{a}_{ii}d_i + \max\left\{0, -\underline{a}_{ii}d_i + \sum_{k\neq i}|a|_{ik}d_k\right\} - \sum_{k\neq i}|a|_{ik}d_k. \qquad (12.31)$$

Two cases must now be discussed.

CASE 1: $-\underline{a}_{ii}d_i + \sum_{k\neq i} |a|_{ik}d_k > 0$

Equation (12.31) becomes $(\langle A_{\mathcal{L}}\rangle \boldsymbol{d})_i = 0$.

CASE 2: $-\underline{a}_{ii}d_i + \sum_{k\neq i} |a|_{ik}d_k \leq 0$.

Equation (12.31) becomes $(\langle A_{\mathcal{L}}\rangle \boldsymbol{d})_i = \underline{a}_{ii}d_i - \sum_{k\neq i} |a|_{ik}d_k \geq 0$.

This proves that $\langle A_{\mathcal{L}}\rangle \boldsymbol{d} \geq 0$. Let $(\lambda, \boldsymbol{x})$ be any eigenpair of $A_{\mathcal{L}}$ so that $A_{\mathcal{L}}\boldsymbol{x} = \lambda \boldsymbol{x}$. Since $A_{\mathcal{L}}$ is symmetric, all its eigenpairs are real and the eigenvector $\boldsymbol{x}$ can be scaled in such a way that $\max_i \left|\frac{x_i}{d_i}\right| = \frac{x_j}{d_j} = 1$, for some j. Then,

$$\begin{aligned}
\lambda d_j &= \lambda x_j = \sum_k a_{\mathcal{L},jk}x_k \\
&\geq |a_{\mathcal{L},jj}||x_j| - \sum_{k\neq j} |a_{\mathcal{L},jk}||x_k| \\
&\geq |a_{\mathcal{L},jj}|d_j - \sum_{k\neq j} |a_{\mathcal{L},jk}|d_k \geq 0.
\end{aligned}$$

Therefore, $\lambda \geq 0$.

Remark 12.4.3 The choice of the non-negative vector $\boldsymbol{d}$ in Theorem 12.4.7 is arbitrary. If all its elements are set to 1, the α vector becomes:

$$\alpha_i = \max\left\{0, -\frac{1}{2}\left(\underline{a}_{ii} - \sum_{j\neq i} |a|_{ij}\right)\right\},$$

where the second term is the expression used in Method I.1 for the extended Gerschgorin theorem. As a result, the elements of the non-uniform diagonal shift matrix obtained using Method II.1 with $d_i = 1\ \forall i$ are less than or equal to the elements of the uniform diagonal shift matrix of Method I.1. A second choice of $\boldsymbol{d}$ is based on the variable ranges. If $\boldsymbol{d} = \boldsymbol{x}^U - \boldsymbol{x}^L$, the off-diagonal contributions to the value of α_i are divided by $x_i^U - x_i^L$. This reflects the fact that variables with a wide range have a larger effect on the quality of the underestimator than variables with a smaller range. Similar scaling can be used for all Type I methods. The interval Hessian matrix $[A]$ is replaced by the matrix $[B]$ with $B_{ik} = d_iA_{ik}d_k$, where $d_i = x_i^U - x_i^L$. With the resulting α value, α_B, one α_i parameter is then computed for each variable using the expression $\alpha_i = \alpha_B/d_i^2$.

12.4.4.2 Iterative Methods.

Method II.2: H-Matrix Method In this method, the properties of H-matrices are used to obtain a valid non-uniform diagonal shift matrix.

Theorem 12.4.8 *Consider the symmetric interval matrix $[G]$ and its modified midpoint matrix $\tilde{G}_M$. Let $C = \tilde{G}_M^{-1}$. If*

- *$\tilde{G}_M$ is positive definite, and*
- *the pre-conditioned matrix $C[G]$ is an H-matrix,*

then $[G]$ is positive definite.

Proof. If $\tilde{G}_M$ is positive definite, all its eigenvalues are positive and the matrix C is regular. If $C[G]$ is an H-matrix, it is regular and hence 0 is not an eigenvalue of $[G]$. The eigenvalues of an interval matrix are a continuous function of the matrix elements. Thus if there exists $G \in [G]$ such that at least one of the eigenvalues of G, or equivalently CG, is negative, and $\tilde{G}_M \in [G]$ has positive eigenvalues, there must exist a matrix G' on the path between G and $\tilde{G}_M$ such that at least one eigenvalue of CG' is equal to 0. This contradicts the H-matrix property of $C[G]$. If the two conditions of Theorem 12.4.8 are met, $[G]$ must therefore be positive definite.

This theorem can be used for the identification of an appropriate diagonal shift matrix Δ for the interval matrix $[A]$ by setting $[G] = [A] + 2\Delta$. The modified midpoint matrix of $[G]$ is then $\tilde{A}_M + \Delta$, where $\tilde{A}_M$ is the modified midpoint matrix of $[A]$. The conditions for the positive definiteness of the interval matrix $[A] + 2\Delta$ are then

(i) $\tilde{A}_M + \Delta$ is positive definite, and

(ii) the pre-conditioned matrix $C[A] + 2C\Delta$ is an H-matrix, where the matrix C takes the form $C = \left(\tilde{A}_M + \Delta\right)^{-1}$.

An iterative approach has been proposed by Adjiman *et al.* (1998a) that can be used to compute a matrix Δ which satisfies the above conditions.

Note that if no matrix Δ, such that $C[A] + 2C\Delta$ is an H-matrix, has been identified after a fixed number of iterations, the matrix Δ^E is returned. As a result, the H-matrix method is at least as accurate as the E-matrix method. It is also more computationally expensive.

Method II.3: Minimization of Maximum Separation Distance Since the maximum separation distance between the original function and its underestimator represents a metric of the quality of the underestimator, this method aims to derive a non-uniform diagonal shift matrix Δ which is optimal with respect to d_{max}. This goal can be expressed as an optimization problem of the form

$$\begin{aligned} \min_{\Delta, \boldsymbol{x}} \quad & (\boldsymbol{x}^U - \boldsymbol{x}^L)^T \Delta (\boldsymbol{x}^U - \boldsymbol{x}^L) \\ s.t. \quad & H_f(\boldsymbol{x}) + 2\Delta \geq 0 \\ & \boldsymbol{x} \in [\boldsymbol{x}^L, \boldsymbol{x}^U] \end{aligned}$$

where Δ is a diagonal matrix, and $M \geq 0$ means that the matrix M is positive semi-definite.

Due to the nonconvexity of the above problem, the formulation is relaxed to

$$\begin{array}{ll} \min\limits_{\Delta} & (\boldsymbol{x}^U - \boldsymbol{x}^L)^T \Delta (\boldsymbol{x}^U - \boldsymbol{x}^L) \\ s.t. & [H_f] + 2\Delta \geq 0 \end{array}$$

The presence of the interval Hessian matrix in the constraint makes the identification of the solution of this problem difficult. To further simplify it, $[H_f]$ can be replaced by a real matrix whose minimum eigenvalue is smaller than the minimum eigenvalue of $[H_f]$. The lower bounding Hessian L defined in Theorem 12.4.5 is a natural choice and the maximum distance minimization problem becomes

$$\begin{array}{ll} \min & (\boldsymbol{x}^U - \boldsymbol{x}^L)^T \Delta (\boldsymbol{x}^U - \boldsymbol{x}^L) \\ s.t. & L + 2\Delta \geq 0 \end{array} \tag{12.32}$$

Remark 12.4.4 Problem (12.32), a semi-definite programming problem, is convex and can therefore be solved to global optimality using interior-point methods which have a polynomial worst-case complexity (Vandenberghe and Boyd, 1996). Because of the use of the lower bounding Hessian matrix, the Δ matrix is rigorously valid but not strictly optimal. As far as the quality of the underestimator is concerned, the minimization of maximum distance method is at least as good as the lower bounding Hessian method but no *a priori* comparison with other methods is possible.

12.5 ILLUSTRATIVE EXAMPLE

The illustrative example is a nonlinear optimization problem in two variables with simple box constraints on the variables.

$$\begin{array}{c} \min\limits_{x,y} f(x,y) = \cos x \sin y - \frac{x}{y^2+1} \\ -1 \leq x \leq 2 \\ -1 \leq y \leq 1 \end{array} \tag{12.33}$$

Three minima were identified with the local optimization software MINOS5.4 (Murtagh and Saunders, 1983) in 1000 runs: $f^1 = -2.02181$ at $(x^1, y^1) = (2, 0.10578)$, $f^2 = -0.99495$ at $(x^2, y^2) = (0.63627, -1)$ and $f^3 = 0.95465$ at $(x^3, y^3) = (-1, 1)$.

The objective function is the sum of two nonlinear terms, shown in Figure 12.1. Two different approaches are available for the derivation of interval Hessian matrices: in *Case 1*, a single Hessian is obtained for the objective function whereas in *Case 2*, a Hessian matrix is derived for each term.

Case 1. Based on the Hessian matrix $H_f(x,y)$ of $f(x,y)$, an interval Hessian family $[H_f]$ which contains $H_f(x,y)$ over the domain of interest is obtained.

$$H_f(x,y) = \begin{pmatrix} -\cos x \sin y & -\sin x \cos y + \frac{2y}{(y^2+1)^2} \\ -\sin x \cos y + \frac{2y}{(y^2+1)^2} & -\cos x \sin y + \frac{2x(y^2+1)^2 - 8xy^2(y^2+1)}{(y^2+1)^4} \end{pmatrix}$$

is such that

$$H_f(x,y) \subseteq [H_f] = \begin{pmatrix} [\text{-}0.84148,\ 0.84148] & [\text{-}3.00000,\ 2.84148] \\ [\text{-}3.00000,\ 2.84148] & [\text{-}40.84148,\ 32.84148] \end{pmatrix}$$

for $-1 \leq x \leq 2$ and $-1 \leq y \leq 1$. The second-order derivatives in $H_f(x,y)$ were generated by the automatic differentiation package built in the implementation of the αBB algorithm. Although the expression for the second-order derivative with respect to y could be simplified further, leading to more accurate interval calculations, it has been kept as is to ensure accurate representation of the algorithm's performance. Since $[H_f]$ was obtained through natural interval extensions, it is not the smallest interval Hessian family which contains $H_f(x,y)$. If the exact minimum and maximum values of the Hessian elements are calculated through global optimization, the smallest achievable intervals can be identified. Following this procedure, it was found that the optimal interval Hessian matrix is in fact

$$[H_f]^* = \begin{pmatrix} [\text{-}0.84148,\ 0.84148] & [\text{-}1.52288, 1.38086] \\ [\text{-}1.52288, 1.38086] & [\text{-}2.00608, 4.00181] \end{pmatrix}.$$

The largest overestimate occurs for the most nonlinear term, the second-order derivative with respect to y.

Case 2. The trigonometric term (Term A) and the fractional term (Term B) are treated separately.
Term A : The Hessian matrix $H_A(x,y)$ and the interval Hessian family $[H_A]$ are given by

$$H_A(x,y) = \begin{pmatrix} -\cos x \sin y & -\sin x \cos y \\ -\sin x \cos y & -\cos x \sin y \end{pmatrix}$$

$$[H_A] = \begin{pmatrix} [\text{-}0.84148,\ 0.84148] & [\text{-}1.00000, 0.84148] \\ [\text{-}1.00000,\ 0.84148] & [\text{-}0.84148,\ 0.84148] \end{pmatrix}.$$

In this case, $[H_A]$ is the smallest interval family that contains $H_A(x,y)$.
Term B : The Hessian matrix $H_B(x,y)$ and the interval Hessian family $[H_B]$ are given by

$$H_B(x,y) = \begin{pmatrix} 0 & \frac{2y}{(y^2+1)^2} \\ \frac{2y}{(y^2+1)^2} & \frac{2x(y^2+1)^2-8xy^2(y^2+1)}{(y^2+1)^4} \end{pmatrix}$$

and

$$[H_B] = \begin{pmatrix} [0,0] & [-2,2] \\ [-2,2] & [-40,32] \end{pmatrix}.$$

The optimal interval Hessian matrix is

$$[H_B]^* = \begin{pmatrix} [0.00000,0.00000] & [-0.64952, 0.64952] \\ [-0.64952, 0.64952] & [-2.00000, 4.00000] \end{pmatrix}.$$

In the following sections, every proposed approach for the calculation of α parameters for $f(x,y)$ will be applied to the illustrative example in order to gather the following information:

- Value of α in the initial domain $(x,y) \in [-1,2] \times [-1,1]$.
- Maximum separation distance d_{max} between $f(x,y)$ and its underestimator.
- Number of iterations required for convergence with a relative tolerance of 10^{-3}.

These will be obtained for *Case 1* and *Case 2*. Since the CPU time for this problem is less than 1 second, it cannot be used for a meaningful comparison of the different methods.

12.5.1 Illustration of Uniform Diagonal Shift Matrix

The minimum eigenvalues for the exact Hessian matrices $H_f(x,y)$, $H_A(x,y)$ and $H_B(x,y)$ are compared to the minimum eigenvalues of the corresponding interval Hessian matrices.

Entire Function. The exact minimum eigenvalue of $H_f(x,y)$ is $\lambda^e_{min} = -2.3934$ for $(x,y) \in [-1,2] \times [-1,1]$. For $[H_f]$, the minimum eigenvalue is -41.0652 and it is -3.0817 for $[H_f]^*$. Based on Equation (12.17), a value of α of 20.5326 is the best that can be obtained with the uniform diagonal shift approach. According to the exact eigenvalue calculation, a value of 1.1967 would suffice to guarantee the convexity of the underestimator. This illustrates the importance of careful interval calculations during the construction of the interval Hessian matrix.

Term A. The exact minimum eigenvalue of $H_A(x,y)$ is $\lambda^e_{min,A} = -1.0$ and that of $[H_A]$ is -1.84148.

Term B. The exact minimum eigenvalue of $H_B(x, y)$ is $\lambda^e_{min,B} = -2.0$. For $[H_B]$, the minimum eigenvalue is -40.0998 and for $[H_B]^*$, the optimal interval Hessian matrix for Term B, it is -2.17636.

As both x and y participate in the two terms, the overall exact α parameter for each of these variables is derived from the sum of the exact minimum eigenvalues for Term A and Term B ($\lambda^e_{min,A} + \lambda^e_{min,B} = -3.0$). Whereas the exact α is 1.1967 in Case 1, the value obtained in Case 2 is $\alpha = 1.5$. Thus, considering the two terms separately yields a looser underestimator. This observation is not maintained for all of the methods to be presented.

12.5.2 *Illustration of Method I.1: Gerschgorin's Theorem for Interval Matrices*

The application of the extended Gerschgorin theorem to the illustrative example is summarized in Table 12.1. The calculation of one eigenvalue for the entire function (Case 1) and the use of eigenvalues for each of the two terms in the function (Case 2) yield the same results. Because this scheme relies on the addition and subtraction of the matrix elements, decomposition of the function has no effect on the final outcome when all the bounds on the term eigenvalues are negative, as is the case in this example. If, on the contrary, one of the terms in the decomposition is convex, its positive eigenvalues need not be taken into account, as indicated by Equation (12.17). By neglecting to add positive contributions, such an approach would result in a decrease of the overall eigenvalues or an increase in α. It therefore seems appropriate not to decompose the nonlinear terms when using this α calculation procedure. Further, it would appear that for a function known to involve convex and nonconvex terms, the inclusion of the convex terms with the general nonlinear terms could lead to an improvement in the quality of the underestimator. Caution must be exercised as the overestimations resulting from the use of interval arithmetic may cause a failure to identify convexity. If the calculated lower bounds on the positive eigenvalues are negative, the underestimator will be looser than necessary. In addition, decomposition of the nonlinear terms may result in the construction of smaller Hessian matrices and reduce the overall computational effort. No general conclusions can be drawn on the treatment of convex terms when using the extended Gerschgorin theorem.

12.5.3 *Illustration of $\mathcal{O}(n^3)$ Methods*

The modified midpoint matrix is

$$\tilde{H}_{f,M} = \begin{pmatrix} -0.84148 & -0.07926 \\ -0.07926 & -40.84148 \end{pmatrix},$$

while the modified radius matrix is

$$\widetilde{\Delta H}_f = \begin{pmatrix} 0 & 2.92074 \\ 2.92074 & 0 \end{pmatrix}.$$

In this example, both choices of E generate similar results as shown in Table 12.2.

12.5.4 *Illustration of Method I.3: Mori and Kokame's Method*

The lower vertex matrix for the current solution domain is

$$\underline{H_f} = \begin{pmatrix} -0.84148 & -3.0 \\ -3.0 & -40.84148 \end{pmatrix}.$$

The minimum eigenvalue of $\underline{H_f}$ is -41.0652 but that of the midpoint matrix, $H_{f,M}$, is -0.0016. $\lambda_{min}(\underline{H_f}) - \lambda_{min}(H_{f,M})$ is negative and much smaller than the spectral radius of ΔH_f. The Mori and Kokame technique therefore leads to the construction of much looser underestimators than the E matrix approach with $E = \text{diag}(\Delta H_f)$. This is supported by the results reported in Table 12.3.

12.5.5 *Illustration of Method I.4: The Lower Bounding Hessian Method*

The lower bounding Hessian matrix for the illustrative example is

$$L_f = \begin{pmatrix} -3.48355 & -0.07926 \\ -0.07926 & -43.7622 \end{pmatrix}.$$

Its minimum eigenvalue is -43.7624. The results are shown in Table 12.4. In this example, the interval Hessian matrix is such that the widths of the off-diagonal elements are small compared to the lower bounds on the diagonal elements. Furthermore, the midpoint of the off-diagonal intervals, which is used to determine the off-diagonal elements of L, is close to zero. These two factors lead to the construction of a lower bounding matrix which is almost diagonal and whose diagonal entries are almost equal to the lower bounds on the interval diagonal elements. In such a situation, the bound on the minimum eigenvalue is very accurate. This is not the case if the width of the off-diagonal intervals is large, or if their midpoint is not close to zero.

12.5.6 *Illustration of Method I.5: Hertz's Method*

Two vertex matrices are required for problems in two variables. For the illustrative example, they are

$$H_{f,1} = \begin{pmatrix} -0.84148 & -3 \\ -3 & -40.84148 \end{pmatrix} \text{ and } H_{f,2} = \begin{pmatrix} -0.84148 & 2.84148 \\ 2.84148 & -40.84148 \end{pmatrix}.$$

Their minimum eigenvalues are -41.0652 and -41.0423 respectively. The results are shown in Table 12.5. The α parameter calculated with the Hertz method is based on the exact minimum eigenvalue of the interval Hessian matrix. Consequently, any other uniform diagonal shift method relying on the interval Hessian matrix produces α values greater than or equal to the value generated with the Hertz method.

12.5.7 *Illustration of Method II.1 : Scaled Gerschgorin Theorem*

Both choices for vector $\boldsymbol{d}$, $d_i = 1$ and $d_i = x_i^U - x_i^L$, are considered. As the results in Table 12.6 show, the use of a non-uniform shift is a very effective way to reduce the number of iterations and the maximum separation distance between f and its underestimator. For $d_i = 1$, the α corresponding to variable y, α_y, is the same as that obtained using the Gerschgorin bound on the minimum eigenvalue (Method I.1). However, the value α_x is decreased by 91%. This results in a 63% decrease in the maximum separation distance for the initial domain. Although the use of $\boldsymbol{d} = \boldsymbol{x}_i^U - \boldsymbol{x}_i^L = (3, 2)^T$ does not change the required number of iterations, it shifts a larger part of the underestimation to the y variable as its range is smaller than that of x. This results in a small decrease in the maximum separation distance. Finally, as was the case for Method I.1, the results are the same for Case 1 and Case 2.

12.5.8 *Illustration of Method II.2: H-Matrix Method*

As shown in Table 12.7, the overall results for Case 1 are the same as for Method I.2, the E-matrix method with $E = 0$. The H-matrix method consistently fails to identify a diagonal shift matrix better than that of Method I.2. For Case 2, however, the number of required iterations is reduced and a significantly improved non-uniform diagonal shift matrix is generated by the H-matrix method.

12.5.9 *Illustration of Method II.3: Minimization of Maximum Separation Distance*

For the illustrative example,

$$\Delta = \begin{pmatrix} \alpha_x & 0 \\ 0 & \alpha_y \end{pmatrix} \text{ and } L = \begin{pmatrix} -3.48355 & -0.07926 \\ -0.07926 & -43.7622 \end{pmatrix}.$$

The results are shown in Table 12.8. Although the maximum separation distance is explicitly minimized in this method, the results are slightly worse than those given by Method II.1 which uses the scaled Gerschgorin approach. This can be attributed to the use of the lower bounding Hessian matrix rather than the interval Hessian matrix in Method II.3. This method is the only one for which the treatment of the two nonconvex terms as separate entities (Case 2) yields worse results than the analysis of the entire function (Case 1).

12.5.10 *On the Role of Interval Hessian Calculations*

The tractability of all the α calculation methods presented in this chapter is ensured through the use of interval arithmetic to obtain bounds on the second-order derivatives. Since the mutual dependence of the Hessian matrix elements is neglected in such an analysis, a certain degree of inaccuracy is introduced. Regardless of the chosen procedure, the quality of the constructed underestimator depends to a large extent on the accuracy of the intervals generated for each

Hessian matrix element. As was stated in the presentation of the illustrative example (Section 12.5), an *optimal* interval Hessian matrix can be computed by using global optimization to determine the minimum and maximum values each matrix element can take given a bounded solution space. The resulting matrix is then optimal in the sense that the tightest possible bounds have been obtained for each element. However, this optimal matrix cannot be derived for the general case. Wider but guaranteed intervals can be generated using classical interval arithmetic techniques. One of the main characteristics of such an approach is that the final result differs depending on the initial analytical representation of the matrix elements. For instance, if the same term appears in the numerator and the denominator of a division, natural interval extensions will evaluate these quantities independently, most probably leading to an overestimate. This is the case for the small illustrative example used throughout Section 12.5. If the analytical Hessian matrix for Term B is expressed as

$$H_B(x,y) = \begin{pmatrix} 0 & \frac{2y}{(y^2+1)^2} \\ \frac{2y}{(y^2+1)^2} & \frac{2x}{(y^2+1)^2}\left(1 - \frac{4y^2}{y^2+1}\right) \end{pmatrix},$$

the interval Hessian matrix obtained through standard interval arithmetic is now

$$[H_B] = \begin{pmatrix} [0,0] & [-2,2] \\ [-2,2] & [-12,6] \end{pmatrix}.$$

The second-order derivative with respect to y is found to belong to a much smaller interval than that obtained through automatic differentiation. The minimum eigenvalue of the above interval matrix is -12.3246 and the best α value which can be derived by any of the uniform diagonal shift matrix methods is 7.0831. This represents a significant improvement over the value of 20.9707 obtained in Section 12.5.1 with different analytical expressions. Great care should therefore be taken when carrying out automatic differentiation.

12.6 GLOBAL OPTIMIZATION ALGORITHM, αBB

In this section, the αBB global optimization procedure is presented for locating the global minimum solution of (12.1) based on the refinement of converging lower and upper bounds. Lower bounds are obtained through the solution of convex programming problems (12.13) and upper bounds based on the solution of (12.1) with local methods.

As has been discussed earlier, the maximum separation between the generic and bilinear nonconvex terms and their respective convex lower bounding functions is bounded. For generic nonconvex terms this maximum separation is proportional to the square of the diagonal of the rectangular partition element and for bilinear terms it is proportional to the area of the rectangular domain.

Furthermore, as the size of the rectangular domains approaches zero, these maximum separations go to zero as well. This implies that as the current box constraints $[\mathbf{x}^L, \mathbf{x}^U]$ collapse to a point, (i) the maximum separation between the original objective function of (12.1) and its convex relaxation in (12.13) becomes zero, and (ii) by the same argument, the maximum separation between the original constraint set in (12.1) and the one in (12.13) goes to zero as well. This implies that for every positive feasibility tolerance ϵ_f and $\boldsymbol{x}$ there always exists a positive value δ such that by reducing the rectangular region $[\boldsymbol{x}^L, \boldsymbol{x}^U]$ around $\boldsymbol{x}$ so as $\|\boldsymbol{x}^U - \boldsymbol{x}\| \leq \delta$ differences between the feasible region of the original problem (12.1) and its convex relaxation (12.13) become less than ϵ_f. Therefore, any feasible point $\boldsymbol{x}^c$ of problem (12.13) (even the global minimum solution) becomes at least ϵ_f–feasible for problem (12.1) by sufficiently tightening the bounds on $\boldsymbol{x}$ around this point.

The next step, after establishing an upper and a lower bound on the global minimum, is to refine them. This is accomplished by successively partitioning the initial rectangular region into smaller ones. The number of variables along which subdivision is required is equal to the number of variables $\boldsymbol{x}$ participating in at least one nonconvex term in formulation (12.1). The partitioning strategy involves the successive subdivision of a rectangle into two subrectangles by halving on the middle point of the longest side of the initial rectangle (bisection). Therefore, at each iteration a lower bound on the objective function of (12.1) is simply the minimum over all the minima of problem (12.13) in every subrectangle composing the initial rectangle. Therefore, a straightforward (bound improving) way of tightening the lower bound is to halve, at each iteration, only the subrectangle responsible for the infimum of the minima of (12.13) over all subrectangles, according to the rules discussed earlier. This procedure generates a *nondecreasing* sequence for the lower bound. An *nonincreasing* sequence for the upper bound is derived by solving locally the nonconvex problem (12.1) and selecting it to be the minimum over all the previously recorded upper bounds. Clearly, if the single minimum of (12.13) in any subrectangle is greater than the current upper bound we can safely ignore this subrectangle because the global minimum of (12.1) cannot be situated inside it (fathoming step).

Because the maximum separations between nonconvex terms and their respective convex lower bounding functions are bounded and continuous functions of the size of rectangular domain, arbitrarily small ϵ_f feasibility and ϵ_c convergence tolerances are reached for a finite size partition element.

The basic steps of the proposed global optimization algorithm are as follows:

STEP 1 - *Initialization*

A convergence tolerance, ϵ_c, and a feasibility tolerance, ϵ_f, are selected and the iteration counter $Iter$ is set to one. Current variable bounds $\boldsymbol{x}^{L,Iter}, \boldsymbol{x}^{U,Iter}$ for the first iteration are set to be equal to the global ones $\boldsymbol{x}^{LBD} and \boldsymbol{x}^{UBD}$.

Lower and upper bounds LBD, UBD on the global minimum of (12.1) are initialized and an initial current point $\mathbf{x}^{c,Iter}$ is randomly selected.

STEP 2 - *Local Solution of Nonconvex NLP and Update of Upper Bound* G_0^{UBD}

The nonconvex optimization problem (12.1) is solved locally within the current variable bounds $\boldsymbol{x}^{LBD}, \boldsymbol{x}^{UBD}$. If the solution f_{local}^{Iter} of (12.1) is ϵ_f–feasible the upper bound UBD is updated as follows,

$$UBD \;=\; \min\left(UBD, f_{local}^{Iter}\right)$$

STEP 3 - *Partitioning of Current Rectangle*

The current rectangle $\left[\boldsymbol{x}^{L,Iter}, \boldsymbol{x}^{U,Iter}\right]$ is partitioned into the following two rectangles ($r = 1, 2$):

$$\begin{bmatrix} x_1^{L,Iter} & x_1^{U,Iter} \\ \vdots & \vdots \\ x_{l^{Iter}}^{L,Iter} & \dfrac{\left(x_{l^{Iter}}^{L,Iter} + x_{l^{Iter}}^{U,Iter}\right)}{2} \\ \vdots & \vdots \\ x_n^{L,Iter} & x_n^{U,Iter} \end{bmatrix}, \quad \begin{bmatrix} x_1^{L,Iter} & x_1^{U,Iter} \\ \vdots & \vdots \\ \dfrac{\left(x_{l^{Iter}}^{L,Iter} + x_{l^{Iter}}^{U,Iter}\right)}{2} & x_{l^{Iter}}^{U,Iter} \\ \vdots & \vdots \\ x_n^{L,Iter} & x_n^{U,Iter} \end{bmatrix}$$

where l^{Iter} corresponds to the variable with the longest side in the initial rectangle,

$$l^{Iter} \;=\; \arg\max_i \left(x_i^{U,Iter} \;-\; x_i^{L,Iter}\right).$$

STEP 4 - *Update of* $\alpha_{i,k}^j$*'s inside both subrectangles r=1,2*

The positive parameters $\alpha_{i,k}^j(\boldsymbol{x}^{U,Iter}, \boldsymbol{x}^{L,Iter})$ are updated inside both rectangles $r = 1, 2$ by one of the methods presented in section 12.4.

STEP 5 - *Solution of inside both subrectangles r=1,2*

The convex optimization problem (12.13) is solved inside both subrectangles ($r = 1, 2$) using any convex nonlinear solver (e.g., MINOS5.4 (Murtagh and Saunders, 1983)). If a solution $l_{sol}^{r,Iter}$ is less than the current upper bound, UBD then it is stored along with the solution point $\boldsymbol{x}_{sol}^{r,Iter}$.

STEP 6 - *Update Iteration Counter Iter and Lower Bound LBD*
The iteration counter is increased by one,

$$Iter \;\longleftarrow\; Iter \;+\; 1$$

and the lower bound LBD is updated to the minimum solution over the stored ones from previous iterations. Furthermore, the selected solution is erased from the stored set.

$$\begin{aligned} LBD &= l_{sol}^{r',Iter'} \\ \text{where} \quad l_{sol}^{r',Iter'} &= \min_{r,I} l_{sol}^{r,I}, \quad r = 1,2, \quad I = 1,\ldots,Iter-1. \end{aligned}$$

STEP 7 - *Update Current Point $\boldsymbol{x}^{c,Iter}$ and Current Bounds $\boldsymbol{x}^{L,Iter}, \boldsymbol{x}^{U,Iter}$ on $\boldsymbol{x}$*

The current point is selected to be the solution point of the previously found minimum solution in **STEP 6**,

$$\boldsymbol{x}^{c,Iter} = \boldsymbol{x}_{sol}^{r',Iter'}$$

and the current rectangle becomes the subrectangle containing the previously found solution,

$$\left[\boldsymbol{x}^{L,Iter}, \boldsymbol{x}^{U,Iter}\right] = \begin{bmatrix} x_1^{L,Iter'} & x_1^{U,Iter'} \\ \vdots & \vdots \\ x_{lIter'}^{L,Iter'} & \dfrac{\left(x_{lIter'}^{L,Iter'} + x_{lIter'}^{U,Iter'}\right)}{2} \\ \vdots & \vdots \\ x_n^{L,Iter'} & x_n^{U,Iter'} \end{bmatrix}, \quad \text{if} \quad r' = 1$$

$$\left[\boldsymbol{x}^{L,Iter}, \boldsymbol{x}^{U,Iter}\right] = \begin{bmatrix} x_1^{L,Iter'} & x_1^{U,Iter'} \\ \vdots & \vdots \\ \dfrac{\left(x_{lIter'}^{L,Iter'} + x_{lIter'}^{U,Iter'}\right)}{2} & x_{lIter'}^{U,Iter'} \\ \vdots & \vdots \\ x_n^{L,Iter'} & x_n^{U,Iter'} \end{bmatrix}, \quad \text{if} \quad r' = 2.$$

STEP 8 - *Check for Convergence*

If $(UBD - LBD) > \epsilon_c$, then return to **STEP 2** .

Otherwise, ϵ_c–convergence has been reached and the global minimum solution, and solution points are:

$$\begin{aligned} f^* &\longleftarrow f^{c,Iter''} \\ \boldsymbol{x}^* &\longleftarrow \boldsymbol{x}^{c,Iter''} \\ \text{where} \quad Iter'' &= \arg_I \left\{f^{c,I} = UBD\right\}, \quad I = 1,\ldots,Iter. \end{aligned}$$

The mathematical proof that the proposed global optimization algorithm αBB converges to the the global minimum is based on a similar analysis as the box constrained nonlinear optimization problems presented in Chapter 11 (Maranas and Floudas, 1994b).

12.7 GEOMETRICAL INTERPRETATION OF THE αBB APPROACH

The selected example presented in this section involves a single variable, so that attention can be focused on the graphical analysis of the proposed approach. As the solution space is partitioned into several subdomains, the change in α value is recorded, and the improvement in the quality of the lower bounding function is monitored. The piecewise convex underestimator of the overall function, which is generated by constructing one lower bounding function per subspace according to equation (12.10), is plotted for different domain sizes. The exact α value can be obtained by repeatedly solving the NLP from different starting points. Comparison of the values thus calculated with those provided by the interval Hessian method also yields valuable information.

The chosen example is that of the molecular conformation of pseudoethane, an ethane molecule in which all the hydrogen atoms have been replaced by C, N or O atoms. It is taken from Maranas and Floudas (1994a) where the global minimum potential energy conformation of small molecules is studied. The Lennard–Jones potential is expressed in terms of a single dihedral angle.

The potential energy of the molecule is given by

$$
\begin{aligned}
f(t) = \; & \frac{588600}{\left(3\,r_0^2 - 4\,\cos\theta r_0^2 - 2\left(\sin^2\theta\cos(t-\frac{2\pi}{3}) - \cos^2\theta\right)r_0^2\right)^6} \\
& - \frac{1079.1}{\left(3\,r_0^2 - 4\,\cos\theta r_0^2 - 2\left(\sin^2\theta\cos(t-\frac{2\pi}{3}) - \cos^2\theta\right)r_0^2\right)^3} \\
& + \frac{600800}{\left(3\,r_0^2 - 4\,\cos\theta r_0^2 - 2\left(\sin^2\theta\cos(t) - \cos^2\theta\right)r_0^2\right)^6} \\
& - \frac{1071.5}{\left(3\,r_0^2 - 4\,\cos\theta r_0^2 - 2\left(\sin^2\theta\cos(t) - \cos^2\theta\right)r_0^2\right)^3} \\
& + \frac{481300}{\left(3\,r_0^2 - 4\,\cos\theta r_0^2 - 2\left(\sin^2\theta\cos(t+\frac{2\pi}{3}) - \cos^2\theta\right)r_0^2\right)^6} \\
& - \frac{1064.6}{\left(3\,r_0^2 - 4\,\cos\theta r_0^2 - 2\left(\sin^2\theta\cos(t+\frac{2\pi}{3}) - \cos^2\theta\right)r_0^2\right)^3}
\end{aligned}
$$

where r_0 is the covalent bond length ($1.54\mathring{A}$), θ is the covalent bond angle (109.5^o), and t is the dihedral angle ($0 \leq t \leq 2\pi$). Note that the contribution from 1–2 and 1–3 atom interactions have not been included in the above formulation since bond lengths and bond angles are fixed.

It can be seen from Figure 12.2 this highly nonlinear function exhibits three minima in the range $[0, 2\pi]$. The two local minima occur at $t = 61.42^o$ and $t = 296.12^o$, where the potential energy takes on values of -0.7973 kcal/mol and -1.0399 kcal/mol respectively. The global minimum of -1.0711 kcal/mol is observed at $t = 183.45^o$.

Table 12.9 shows the exact and computed α's for the first four partitions of the solution space, while Table 12.10 summarizes the results for subsequent levels. The accuracy of the calculated α values is seen to improve rapidly as the size of the search domain is reduced, confirming the importance of supplying tight variables bounds. The numerical value of α is not sufficient to assess the performance of the method since the convergence of the algorithm is largely determined by the separation distance between the function and its underestimator, which depends both on α and the size of the domain. Tables 12.11 and

12.12 give the maximum separation distance for the exact and interval Hessian calculations. Although the former provide much tighter underestimators at the initial levels of the tree, the latter decrease at a faster rate. By plotting the piecewise convex lower bounding function at every level in the tree, the rapid improvement in the quality of the underestimator can be visualized (Figure 12.3, Figure 12.4).

When the αBB algorithm is used to identify the global optimum solution, convergence to within 10^{-6} is achieved in 21 iterations and 1.1 CPU seconds on an HP-9000/730. Figure 12.5 shows the branch and bound tree as explored during the αBB run. Although the search is initially breadth–first, most nodes on level 5 are fathomed and the solution is quickly identified as the underestimator tightens. For an unconstrained optimization problem in one variable such as this one, the exact eigenvalue at each level can be identified with the help of graphical analysis.

Due to the use of interval arithmetic in the Hessian matrix method, it is important to derive a mathematical expression for the second–order derivatives which will reduce the occurence of interval overestimation. In the case of pseudoethane, it is interesting to note that better estimates of the eigenvalue are generated if the potential function is not differentiated in the dihedral angle space but in the r space, where r is the distance between two atoms. Once the r–space Hessian has been obtained, it can be expressed in terms of the dihedral angle through a transformation of coordinates. At the first level, for example, a value of 620 is obtained for α, as compared with approximately 38,000 using differentiation in the dihedral angle space. The algorithm terminates after 15 iterations and 0.9 CPU seconds. In some instances where explicit expressions for the eigenvalues are not available, the user may therefore be able to provide expressions superior to those obtained through direct differentiation. Using the exact minimum eigenvalue at each iteration, the solution is found in 10 iterations, with an initial α value of 6.7. Computing an accurate interval Hessian matrix is of the utmost importance.

Summary

In this chapter, we discussed the theoretical issues of the αBB global optimization approach for general constrained twice differentiable nonlinear optimization problems. Important properties that result in the underestimation of special structure terms such as bilinear, trilinear, fractional, fractional trilinear, as well as general structure nonlinear terms were presented. Several methods for the rigorous calculation of the α parameters based on interval Hessian approximations were discussed using a small illustrative example. The geometrical interpretation of the αBB global optimization approach was presented via an example arising in computational chemistry.

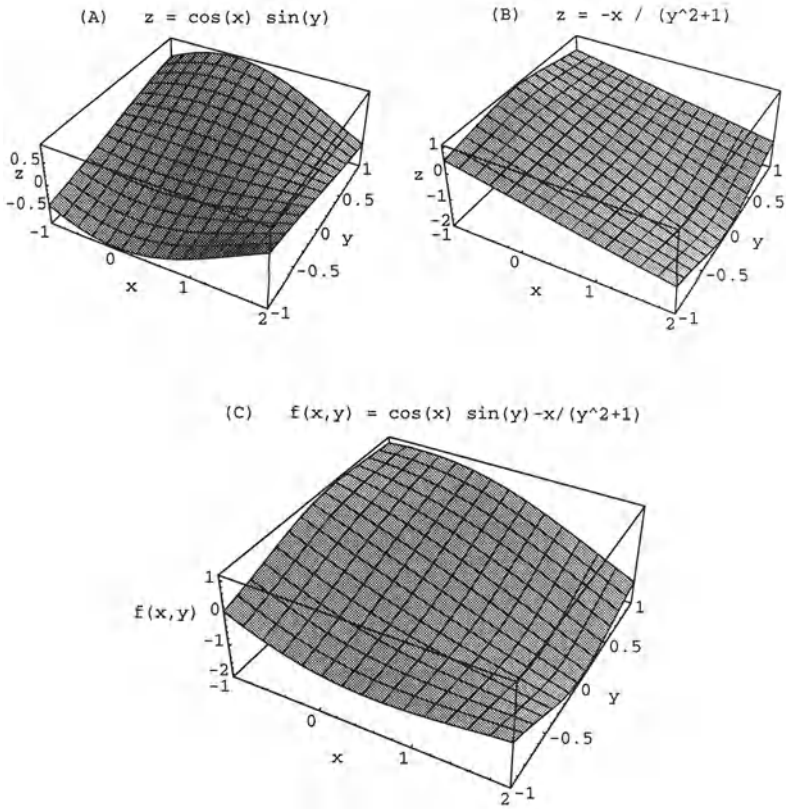


Figure 12.1. Illustrative Example – The objective function (C) is the sum of two terms (A and B)

Case	λ_{min}	α	d_{max}	Iterations
1	-43.85	21.93	71.24	19
2	-43.85	21.93	71.24	20

Table 12.1. Results for the illustrative example using the Gerschgorin theorem.

	$E = 0$				$E = \text{diag}(\Delta H_f)$			
Case	λ_{min}	α	d_{max}	Iter	λ_{min}	α	d_{max}	Iter
1	-43.77	21.89	71.11	18	-45.04	22.52	73.18	18
2	-43.85	21.93	71.24	18	-45.94	22.97	74.64	18

Table 12.2. Results for the illustrative example using the E-matrix method.

Case	λ_{min}	α	d_{max}	Iterations
1	-131.14	65.57	213.09	31
2	-133.65	66.83	217.18	32

Table 12.3. Results for the illustrative example using the Mori and Kokame method.

Case	λ_{min}	α	d_{max}	Iterations
1	-43.77	21.89	71.11	18
2	-43.85	21.93	71.24	18

Table 12.4. Results for the illustrative example using the lower bounding Hessian method.

Case	λ_{min}	α	d_{max}	Iterations
1	-41.07	20.54	66.73	18
2	-41.95	20.98	68.15	18

Table 12.5. Results for the illustrative example using the Hertz method.

Case	$d_i = 1$				$d_i = x_i^U - x_i^L$			
	α_x	α_y	d_{max}	Iter	α_x	α_y	d_{max}	Iter
1	1.93	21.93	26.24	13	1.43	22.68	25.87	13
2	1.93	21.93	26.24	13	1.43	22.68	25.87	13

Table 12.6. Results for the illustrative example using the scaled Gerschgorin method.

Case	α_x	α_y	d_{max}	Iterations
1	21.89	21.89	71.11	18
2	0.93	21.93	23.99	16

Table 12.7. Results for the illustrative example with the H-matrix method.

Case	α_x	α_y	d_{max}	Iterations
1	1.91	21.95	26.23	14
2	1.91	21.95	26.23	16

Table 12.8. Results for the illustrative example with minimization of maximum distance.

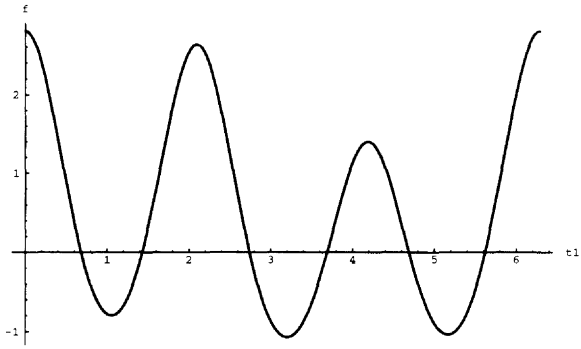


Figure 12.2. Potential energy of pseudoethane as a function of the dihedral angle.

Level	Search Domain	Exact α	Calculated α
1	$[0, 2\pi]$	10.7	$3.8\ 10^4$
2	$[0, \pi]$	10.7	$9.5\ 10^3$
2	$[\pi, 2\pi]$	10.7	$8.6\ 10^3$
3	$[0, \frac{\pi}{2}]$	10.7	944.2
3	$[\frac{\pi}{2}, \pi]$	10.4	298.9
3	$[\pi, \frac{3\pi}{2}]$	7.9	262.0
3	$[\frac{3\pi}{2}, 2\pi]$	10.7	869.2
4	$[0, \frac{\pi}{4}]$	10.7	56.8
4	$[\frac{\pi}{4}, \frac{\pi}{2}]$	0	93.5
4	$[\frac{\pi}{2}, \frac{3\pi}{4}]$	10.4	41.3
4	$[\frac{3\pi}{4}, \pi]$	5.5	87.0
4	$[\pi, \frac{5\pi}{4}]$	4.0	75.8
4	$[\frac{5\pi}{4}, \frac{3\pi}{2}]$	7.9	34.5
4	$[\frac{3\pi}{2}, \frac{7\pi}{4}]$	0	84.4
4	$[\frac{7\pi}{4}, 2\pi]$	10.7	56.7

Table 12.9. Exact and calculated α s for the pseudoethane potential energy function

Level	Interval width	Range for exact α	Range for Calculated α
5	$\frac{\pi}{8}$	0 – 10.7	10.2 – 23.5
6	$\frac{\pi}{16}$	0 – 10.7	2.8 – 14.0
7	$\frac{\pi}{32}$	0 – 10.7	0.0 – 10.7
8	$\frac{\pi}{64}$	0 – 10.7	0.0 – 10.7

Table 12.10. Range for exact and calculated α s for the pseudoethane potential energy function

Level	Search Domain	$\max(f - \mathcal{L}^*)$	$\max(f - \mathcal{L})$
1	$[0, 2\pi]$	105.6	$3.8\ 10^5$
2	$[0, \pi]$	26.4	$2.3\ 10^4$
2	$[\pi, 2\pi]$	26.4	$2.1\ 10^4$
3	$[0, \frac{\pi}{2}]$	6.6	582.4
3	$[\frac{\pi}{2}, \pi]$	6.4	184.4
3	$[\pi, \frac{3\pi}{2}]$	4.9	161.6
3	$[\frac{3\pi}{2}, 2\pi]$	6.6	536.2
4	$[0, \frac{\pi}{4}]$	1.7	8.8
4	$[\frac{\pi}{4}, \frac{\pi}{2}]$	0	14.4
4	$[\frac{\pi}{2}, \frac{3\pi}{4}]$	1.6	6.4
4	$[\frac{3\pi}{4}, \pi]$	0.8	13.4
4	$[\pi, \frac{5\pi}{4}]$	0.6	11.7
4	$[\frac{5\pi}{4}, \frac{3\pi}{2}]$	1.2	5.3
4	$[\frac{3\pi}{2}, \frac{7\pi}{4}]$	0	13.0
4	$[\frac{7\pi}{4}, 2\pi]$	1.7	8.7

Table 12.11. Maximum separation distance between function and underestimator. $\mathcal{L}^*$ denotes the exact lower bounding function while $\mathcal{L}$ corresponds to the Hessian matrix calculation

Level	Interval width	Range of $\max(f - \mathcal{L}^*)$	Range of $\max(f - \mathcal{L})$
5	$\frac{\pi}{8}$	0 – 0.413	0.393 – 0.906
6	$\frac{\pi}{16}$	0 – 0.103	0.027 – 0.135
7	$\frac{\pi}{32}$	0 – 0.026	0 – 0.026
8	$\frac{\pi}{64}$	0 – 0.006	0 – 0.006

Table 12.12. Range of maximum separation distance between function and underestimator. $\mathcal{L}^*$ denotes the exact lower bounding function while $\mathcal{L}$ corresponds to the Hessian matrix calculation

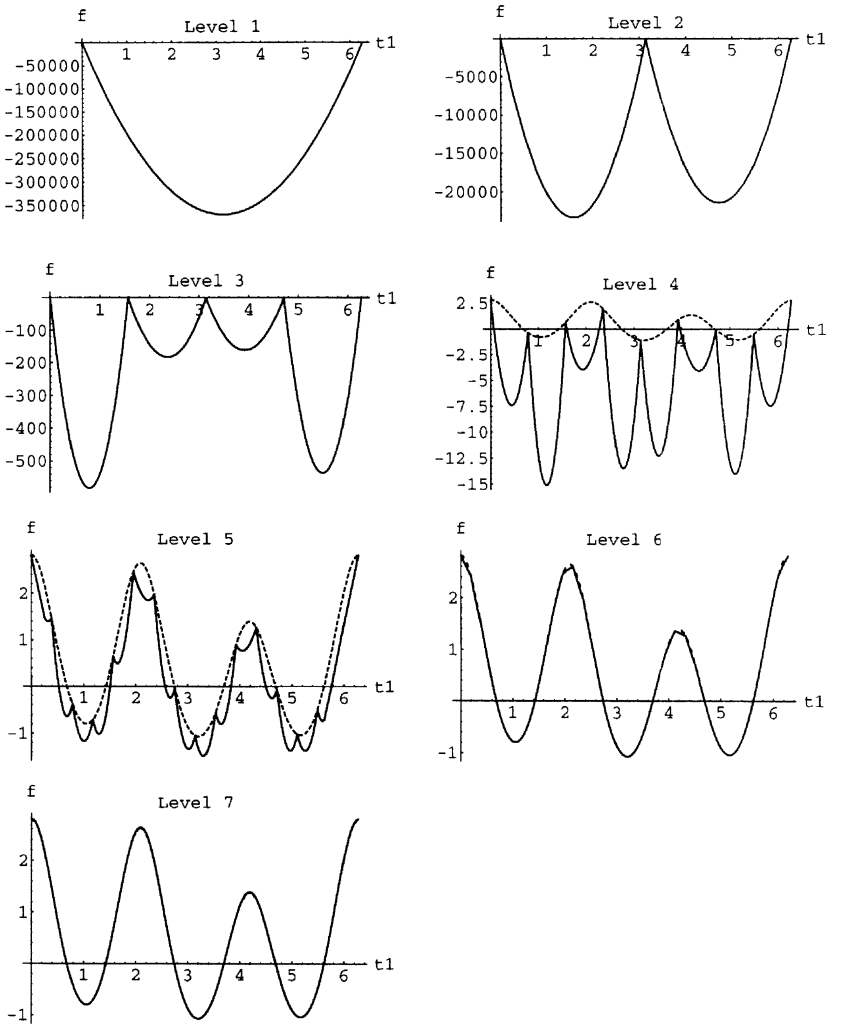


Figure 12.3. Function and underestimator at different levels of the branch–and–bound tree using calculated α values.

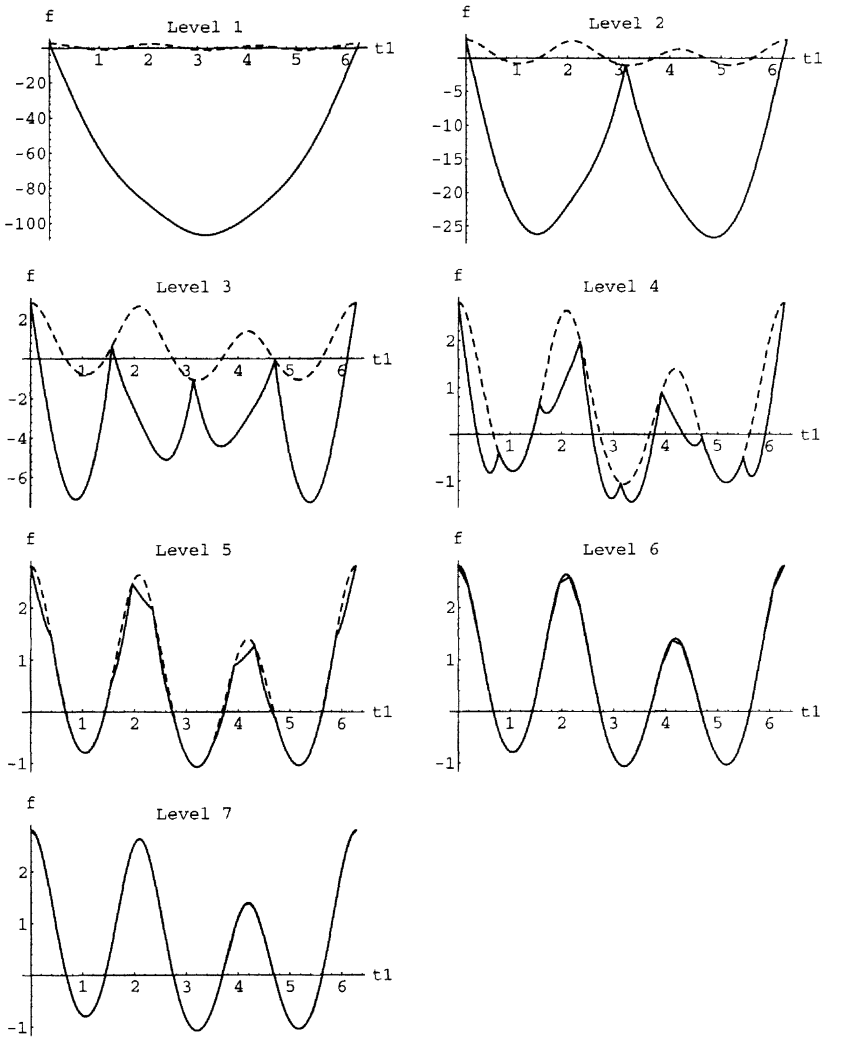


Figure 12.4. Function and underestimator at different levels of the branch–and–bound tree using exact α values.

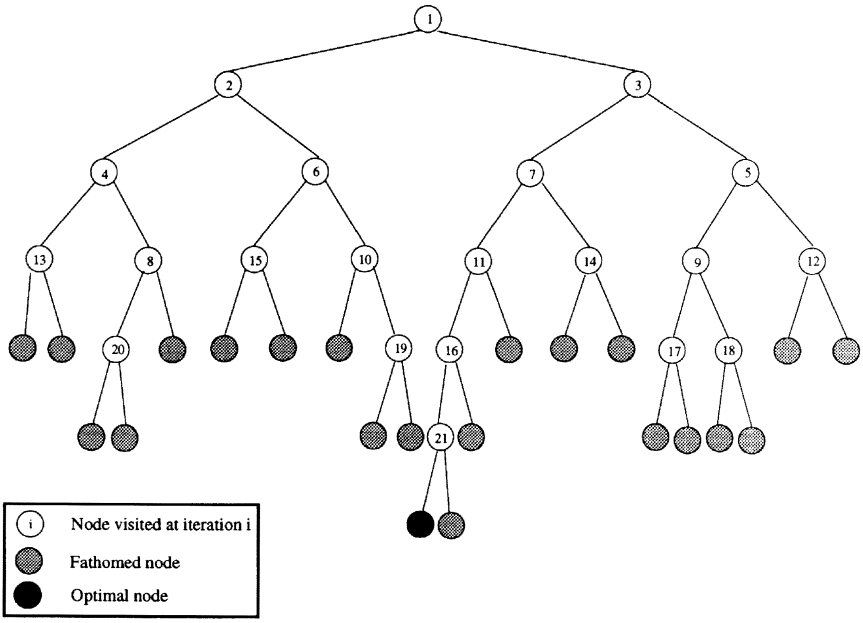


Figure 12.5. αBB branch and bound tree for pseudoethane example, using calculated α values.

13 COMPUTATIONAL STUDIES OF THE αBB APPROACH

In this chapter, we discuss the algorithmic issues of the αBB global optimization approach, along with its implementation and a variety of computational studies. Section 13.1 focuses on the important issues of branching strategies and variable bound updates. Section 13.2 describes the implementation of the αBB. Finally, section 13.3 presents extensive computational studies on literature, engineering design and stability problems. The material presented in this chapter is based on the work of Adjiman *et al.* (1998b).

13.1 ALGORITHMIC ISSUES AND DEVELOPMENT OF THE αBB

Based on the theoretical advances presented in Chapter 12, in this section we present important issues in the algorithmic development of the αBB global optimization approach. The structure of the algorithm is shown in Figure 13.1 and has the following features:

- several alternatives for the construction of the lower bounding problem depending on the nature of the terms,
- an interface with MINOPT (Schweiger and Floudas, 1997), which contains a collection of local NLP, MILP and MINLP solvers,
- the capability to derive Hessian matrices or characteristic polynomials via automatic differentiation,

C. A. Floudas, *Deterministic Global Optimization*
© Springer Science+Business Media Dordrecht 2000

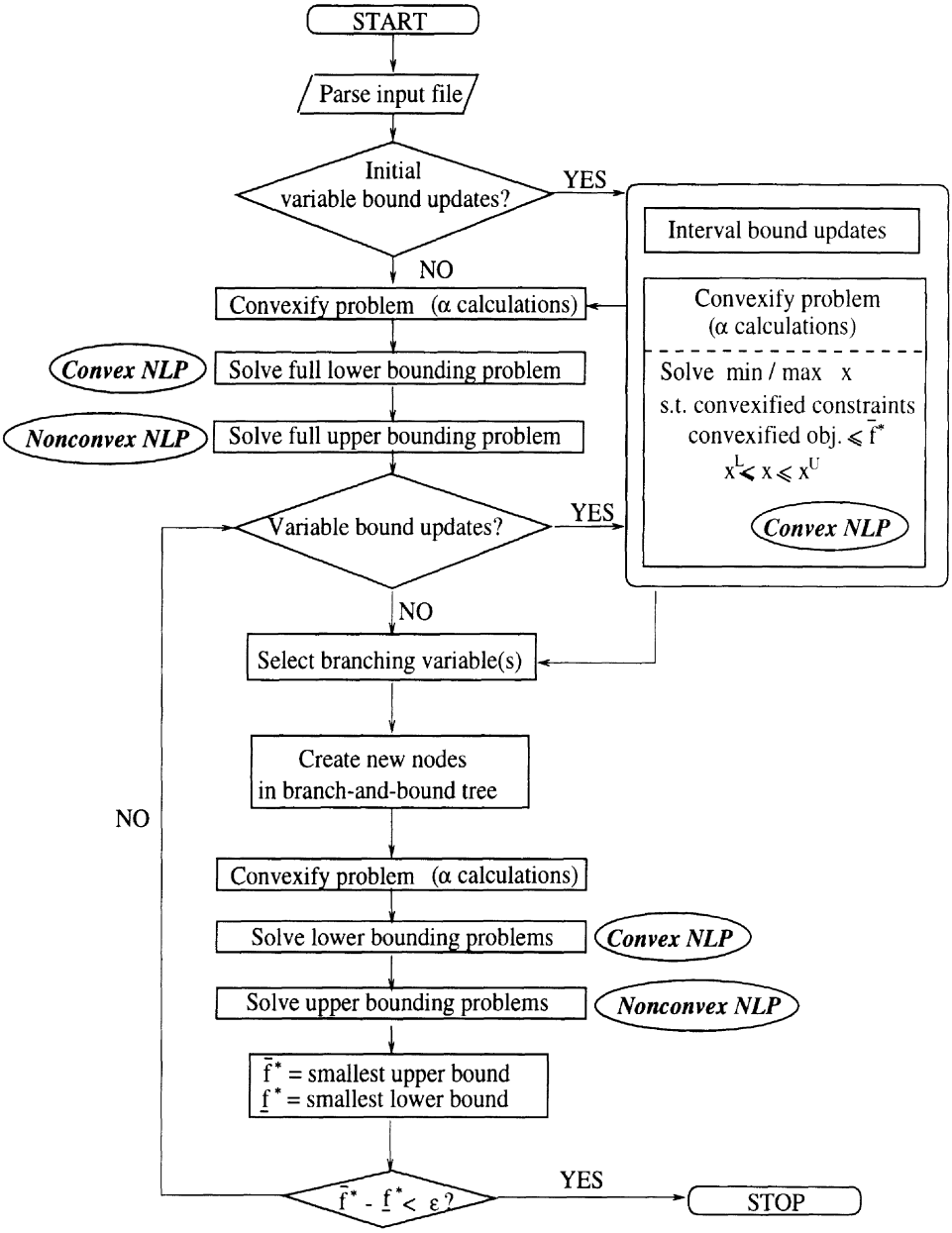


Figure 13.1. Flowchart for the αBB algorithm

- the ability to generate interval Hessian matrices and to perform interval arithmetic,
- methods for calculation of α using the approaches presented in Chapter 12,
- several branching strategies,
- several variable bounds update strategies.

In the sequel, we discuss the branching and variable bounding rules which are used in order to enhance the performance of the αBB algorithm.

13.1.1 Branching Strategies

The branching step of any branch-and-bound algorithm often has a significant effect on the rate of convergence. This is especially true of the αBB algorithm as the quality of the underestimator depends on the variable bounds in a variety of ways. For instance, if a variable participates only linearly in the problem, branching on it will not have any effect on the accuracy of the convex lower bounding functions. On the other hand, reducing the range of a variable raised to a high power is likely to result in much tighter underestimating problems. To take advantage of these observations, the implementation of the αBB algorithm offers some choice in the branching strategies. Four alternatives are currently available:

1. Use k-section on all or some of the variables.
2. Use a measure of the quality of each term's underestimator, based on the maximum separation distance between term and underestimator.
3. Use a measure of the quality of each term's underestimator, based on the separation distance at the optimum point.
4. Use a measure of the overall influence of each variable on the quality of the lower bounding problem.

13.1.1.1 Strategy 1. By default, bisection on all variables is used. The user can however specify a restricted list of variables on which to branch. The interval to be bisected is determined through the '*least reduced axis*' rule whose application involves the calculation of a 'current range to original range ratio', r_i, for each variable:

$$r_i = \frac{x_i^U - x_i^L}{x_{i,0}^U - x_{i,0}^L} \tag{13.1}$$

where $x_{i,0}^U$ and $x_{i,0}^L$ are, respectively, the upper and lower bound on variable x_i at the first node of the branch-and-bound tree, and x_i^U and x_i^L are, respectively, the upper and lower bound on variable x_i at the current node of the tree. The variable with the largest r_i is selected for branching.

13.1.1.2 Strategy 2. While the second branching option requires additional computational effort, it results in a significant improvement in the convergence time for difficult problems. A different measure μ is defined for each type of underestimator in order to facilitate the assessment of the quality of the lower bounding function. The maximum separation distance between the underestimator and the actual term at the optimal solution of the lower bound problem is one measure of the quality of the convex approximation.
In Chapter 12 we have seen that for a bilinear term xy, the maximum separation distance μ_b is

$$\mu_b = \frac{(x^U - x^L)(y^U - y^L)}{4}. \tag{13.2}$$

For a fractional term x/y, the maximum separation distance is given by equation (12.7) of Chapter 12.
For a univariate concave term $ut(x)$, the maximum separation distance can be expressed as an optimization problem so that μ_u is given by

$$\mu_u = -\min_{x^L \leq x \leq x^U} -ut(x) + ut^L(x) \tag{13.3}$$

where $ut^L(x)$ is the linearization of $ut(x)$ around x^L. Since $ut(x)$ is concave and $ut^L(x)$ is linear, the above optimization problem is convex and can therefore be solved to global optimality.
For a general nonconvex term, the maximum separation distance was derived by Maranas and Floudas (1994b) and the term measure is

$$\mu_\alpha = \frac{1}{4}\sum_i \alpha_i (x_i^U - x_i^L)^2 \tag{13.4}$$

where the α_i's are defined such that the underestimator in Equation (12.10) is convex.

Given a node to be partitioned, the values of μ_b, μ_u and μ_α are calculated for each term in accordance with its type. The term which appears to have the worst underestimator (i.e., the largest μ), is then used as a basis for the selection of the branching variable. Out of the set of variables that participate in that term, the one with the '*least reduced axis*', as defined by Equation (13.1), is chosen for k-section. With this strategy, the influence of variable bounds on the quality of the underestimators is taken into account, and hence this adaptive branching scheme ensures the effective tightening of the lower bounding problem from iteration to iteration.

13.1.1.3 Strategy 3. This strategy is a variant of Strategy 2. Instead of computing the maximum separation distance between a term and its underestimator, their separation distance at the *optimum solution* of the lower bounding problem is used. Thus, the bilinear term measure is now

$$\mu_b = |x^* y^* - w^*| \tag{13.5}$$

where w is the variable which has been substituted for the bilinear term xy in order to construct its convex envelope (Adjiman *et al.*, 1998a) and the * superscript denotes the value of the variable at the solution of the current lower bounding problem. The measures μ_t, μ_f and μ_{ft}, for trilinear, fractional and fractional trilinear terms, are obtained in a similar fashion.
The measure for univariate term $ut(x)$ is

$$\mu_u = ut(x^*) - ut^L(x^*). \tag{13.6}$$

Finally, the general nonconvex term measure is

$$\mu_\alpha = \sum_{i=1}^{n} \alpha_i (x_i^U - x_i^*)(x_i^* - x_i^L). \tag{13.7}$$

13.1.1.4 Strategy 4. The fourth branching procedure takes the approach of Strategy 3 one step further by considering the overall influence of each variable on the convex problem. After the relevant measures μ_b, μ_t, μ_f, μ_{ft}, μ_u and μ_α have been calculated for every term, a measure μ_v of each variable's contribution may be obtained as follows:

$$\mu_v^i = \sum_{j \in B_i} \mu_b^j + \sum_{j \in T_i} \mu_t^j + \sum_{j \in F_i} \mu_f^j + \sum_{j \in FT_i} \mu_{ft}^j + \sum_{j \in U_i} \mu_u^j + \sum_{j \in N_i} \mu_\alpha^j \tag{13.8}$$

where μ_v^i is the measure for the ith variable; B_i is the index set of the bilinear terms in which the ith variable participates, μ_b^j is the measure of the jth bilinear term; T_i is the index set of the trilinear terms in which the ith variable participates, μ_t^j is the measure of the jth trilinear term; F_i is the index set of the fractional terms in which the ith variable participates, μ_f^j is the measure of the jth fractional term; FT_i is the index set of the fractional trilinear terms in which the ith variable participates, μ_{ft}^j is the measure of the jth fractional trilinear term; U_i is the index set of the univariate concave terms in which the ith variable participates, μ_u^j is the measure of the jth univariate concave term; N_i is the index set of the general nonconvex terms in which the ith variable participates, μ_α^j is the measure of the jth general nonconvex term.

The variable with the largest measure μ_v is selected as the branching variable, and k-section can be performed on it. If two or more variables have the same μ_v, the '*least reduced axis*' test is performed to distinguish them.

As will become apparent in the computational studies, branching strategies 2, 3 and 4 are particularly effective since they take into account the sensitivity of the underestimators to the bounds used for each variable.

13.1.2 Variable Bound Updates

The quality of the convex lower bounding problem can also be improved by ensuring that the variable bounds are as tight as possible. In the current

implementation of the αBB algorithm, variable bound updates can either be performed at the onset of an αBB run or at each iteration.

In both cases, the same procedure is followed in order to construct the bound update problem. Given a solution domain, the convex underestimator for every constraint in the original problem is formulated. The *bound problem* for variable x_i is then expressed as

$$x_i^{L,NEW}/x_i^{U,NEW} = \begin{cases} \min_{\boldsymbol{x}} / \max_{\boldsymbol{x}} & x_i \\ s.t. & \mathcal{G}(\boldsymbol{x}) \leq 0 \\ & \boldsymbol{x}^L \leq \boldsymbol{x} \leq \boldsymbol{x}^U \end{cases} \tag{13.9}$$

where $\mathcal{G}(\boldsymbol{x})$ are the convex underestimators of the constraints (note that we include in them the convex underestimator of the objective function being less or equal than the best upper bound), and the bounds on the variables, $\boldsymbol{x}^L$ and $\boldsymbol{x}^U$ are the best calculated bounds. Thus, once a new lower bound $x_i^{L,NEW}$ on x_i has been computed via a minimization, this value is used in the formulation of the maximization problem for the generation of an upper bound $x_i^{U,NEW}$.

Because of the computational expense incurred by an update of the bounds on all variables, it is often desirable to define a smaller subset of the variables on which this operation is to be performed. The criterion devised for the selection of the branching variables can be used in this instance, since it provides a measure of the sensitivity of the problem to each variable. An option was therefore set up, in which bound updates are carried out only for a fraction of the variables with a non-zero μ_v, as calculated in Equation (13.8).

13.2 IMPLEMENTATION OF THE αBB

The main objectives in the implementation of the αBB were to create an efficient and user-friendly global optimization environment for NLP problems involving twice-differentiable functions, while ensuring that the user interface remains simple and flexible. This section describes the choices made in order to meet these objectives.

All the strategies for branching variable selection and variable bound updates presented in the previous section, as well as the underestimation techniques presented in Chapter 12 have been implemented as part of the αBB package. In addition, an intuitive interface and a user's manual (Adjiman *et al.*, 1997) have been designed.

13.2.1 Problem Input and Pre-Processing

The need for the second-order derivatives of the nonconvex functions in the problem raises practical difficulties in terms of the implementation of the algorithm and the format of the input file. Because the problems solved belong to a very broad class for which no structural assumptions may be made *a priori*, these derivatives must either be supplied by the user or generated automatically. The first option would be very cumbersome. In order to generate the

derivatives automatically, the functions must be available in a code list format (Rall, 1981) which allows systematic differentiation based on elementary rules. Although the functions could be provided by the user in this codified notation, such an approach would result in counter-intuitive problem input. By designing an interface which accepts input in standard mathematical notation, transforms it into the desired form and automatically generates sparse Jacobian and Hessian matrices, both ease of use and flexibility can be achieved. The front-end parser developed as part of the αBB implementation creates this code list, while giving much freedom of notation to the user. This parser can also be used to input user-provided expressions for the second order derivatives when these cannot be obtained automatically. During the parsing phase of an αBB run, the following tasks are carried out:

- Identify variable, function, and parameter names.
- Define linear, bilinear, trilinear, fractional, fractional trilinear, convex, univariate concave, and general nonconvex terms. This determines what types of underestimators are to be used.
- Build code lists for the functions.
- Define bounds on the solution space.
- Gather optional information such as branching strategy, user-defined α values if any.

The choice of all names (variable, function, term, parameter) is left up to the user, with the possibility to use index notation. Many index operations are supported, such as summation, multiplication, enumeration and index arithmetic. The input can therefore be made as explicit or as compact as desired. A sample input file for the CSTR sequence design problem and the formulation of this problem are provided in Adjiman *et al.* (1998b).

Once the parsing phase is completed, a processing stage is initiated in which the code list for each of the lower bounding functions is built using the term information provided in the input file. Sparse Jacobians and Hessian matrices are also generated in the code list format. The sparsity issue is critical not only from the point of view of memory requirements, but also with respect to the diagonal shift matrix computation. The speed of this process is strongly dependent on the number of participating variables for all the methods described in Chapter 12.

13.2.2 User-Specified Underestimators

In some instances, the problem to be solved may involve functions for which no explicit formulation is available. The potential energy function for proteins, for example, is typically calculated using force-field models such as CHARMM (Brooks *et al.*, 1983a), AMBER (Weiner *et al.*, 1986) or the ECEPP/3 model (Némethy *et al.*, 1992). The user can require that these be used for function

evaluations, as long as the Jacobian and a convex underestimator are also provided.

Some functions whose analytical form is entered in the input file may exhibit a special structure for which a tight underestimator is known. This tailored lower bounding function can then be entered in the input file and replace the generic nonconvex term underestimator. Similarly, if values or formulae have been derived for the α parameters, this information can be used by the algorithm, thereby improving its performance.

13.2.3 α Calculations

Two main options are available to the user for the determination of the α values to be used in the lower bounding problem.

Option 1: User-specified expressions supplied in the input file for α computations.

Option 2: Calculation using one of the automatic computation methods described in Chapter 12.

The successful implementation of the second option requires the availability of a rigorous procedure for interval function evaluations. The αBB algorithm is therefore connected to a C++ interval arithmetic library, PROFIL/BIAS, developed by Knüppel (1993). All the interval calculations are carried out using the natural extension form of the functions (Ratschek and Rokne, 1984). The main feature of this form is its simplicity but more accurate results can be obtained using the centered form or the remainder form (Cornelius and Lohner, 1984), at increased computational expense. The nested form is particularly suited for the special case of polynomial functions because of the quality of its results and the ease of computation.

Most of the $\mathcal{O}(n^3)$ methods for the computation of a uniform diagonal shift matrix require the calculation of the minimum and/or maximum eigenvalue of real symmetric matrices. These matrices are usually dense and of small size since only the variables that participate in the nonconvex term to be underestimated are taken into consideration when building the Hessian matrix. Some Netlib routines for the transformation of real symmetric matrices into symmetric tridiagonal matrices using orthogonal similarity transformations and the rational QR method with Newton corrections were translated into C for this purpose. The code was also customized based on the specific requirements of the αBB algorithm.

Finally, the error in the α parameter calculations is closely monitored to guarantee the global optimality of the final solution.

13.3 COMPUTATIONAL STUDIES OF THE αBB APPROACH

In this section we discuss the performance of the αBB algorithm in identifying the global minimum of nonconvex optimization problems and we study the

effects of the different underestimating methods presented in Chapter 12 and the various convergence-enhancing schemes presented in the previous sections. First, problems taken from the literature such as the Haverly pooling problems, linearly constrained concave quadratic problems, and a small but difficult nonlinear constrained problem, are presented, giving insights into the suggested branching and variable range reduction strategies. Second, problems that arise in engineering design are presented. Third, a number of stability of control systems with real parametric uncertainty problems that were also addressed in Chapter 9 are revisited through the αBB approach. All computational results were obtained on a HP9000/730 with a convergence tolerance of 0.001, unless specified otherwise.

13.3.1 Problems Taken from the Literature

Haverly Pooling Problem. This problem is taken from Floudas and Pardalos (1990) and was also studied via the GOP approach, as presented in section 4.2.3 of Chapter 4 where the mathematical formulation is provided. Three instances of the problem, denoted I, II, and III, have been defined based on the values of the parameters (c_1, c_2). These are (16, 100), (16, 600), and (13, 100) respectively. The nonconvexities are entirely due to the presence of bilinear terms, which can be underestimated using their convex envelope as specified in Chapter 12, or as general nonconvex terms, using calculated α values.

The problems were first solved using different options for the variable bound updates and the branching strategy, while using the convex envelope for the bilinear terms. The results are presented in Table 13.1. In addition, results obtained using the most recent implementation of the GOP global optimization algorithm as discussed in Chapter 4, and designed specifically to handle problems containing bilinear, quadratic and polynomial terms, are listed.

As far as the αBB algorithm is concerned, it is clear that Run E is the most promising overall strategy. The addition of variable bounds updates results in a significant improvement of the convergence rate as is revealed by a comparison of Runs A and B. Important insight into the workings of the algorithm can be gained by recording the variables on which branching is performed when different strategies are used. In branching strategy 1 (Runs A and B), each variable is branched on equally and the lower bounds remain constant for several iterations as the variables participating linearly have little effect on the tightness of the underestimating problem. In other runs, branching depends on the term measure μ_b, either directly or through the variable measure μ_v. Since the bilinear terms include three variables, p, P_x and P_y, they are the only branching candidates. This leads to another considerable improvement in performance. With branching strategies 2 and 3, each one of the three variables is branched on equally. With branching strategy 4, however, p is branched on exclusively as it participates in all bilinear terms and its μ_v is always greater than that of P_x or P_y. The performance of the branch-and-bound search is therefore enhanced substantially by appropriately selecting the branching variables.

Run	*I*			*II*			*III*		
	N_i	N_n	CPU	N_i	N_n	CPU	N_i	N_n	CPU
A	73	147	3.16	79	159	4.35	161	323	6.33
B	11	23	2.73	9	19	2.42	15	31	3.17
C	5	11	1.22	2	5	0.62	8	17	1.80
D	5	11	1.19	2	5	0.59	8	17	1.84
E	2	5	0.61	2	5	0.61	3	7	0.81
GOP	12	—	0.22	12	—	0.21	14	—	0.26

Run A : No variables bounds update, branching strategy 1,
Run B : Variable bounds updates at each iteration, branching strategy 1,
Run C : Variable bounds updates at each iteration, branching strategy 2,
Run D : Variable bounds updates at each iteration, branching strategy 3,
Run E : Variable bounds updates at each iteration, branching strategy 4.
Note N_i is the number of iterations and N_n is the number of expanded nodes.

Table 13.1. Comparison of computational results using various strategies.

Using the same branching and bounding options as in Run E, the problems were then solved by treating the bilinear terms as general nonconvex terms. For bilinear terms, the Hessian matrix is independent of the variables and no interval calculations are required. The minimum eigenvalue required by all uniform diagonal shift matrix methods is therefore easily obtained. Table 13.2 compares the performance of the algorithm using the convex envelope ("Linear") and α-based underestimation. The row labeled "Convex I" reports results obtained using a uniform diagonal shift matrix (a single α value per term), and "Convex II" was obtained using the scaled Gerschgorin theorem method which generates one α per variable in a nonconvex term. The use of α leads to looser lower bounding functions than the convex envelope. Moreover, it requires the solution of a convex NLP for the generation of a lower bound, whereas a linear program is constructed when the convex envelope is used. As a result, both computation time and number of iterations increase significantly. The exploitation of the special structure of bilinear terms is thus expected to provide the best results in most cases. For very large problems, however, the introduction of additional variables and constraints may become prohibitive and convex underestimation becomes more appropriate (Harding and Floudas, 1997).

Concave Quadratic Optimization Problems. This set of problems, taken from Floudas and Pardalos (1990), will be used to show the increased efficiency of the algorithm when appropriate strategies are being employed to select the variables whose bounds will be updated.
The general form of the problems is as follows :

Underestimator	I			II			III		
	N_i	N_n	CPU	N_i	N_n	CPU	N_i	N_n	CPU
Linear	2	5	0.61	2	5	0.61	3	7	0.81
Convex I	24	49	5.66	27	55	7.23	18	37	4.53
Convex II	31	63	6.72	23	47	6.48	14	29	3.60

Table 13.2. Computational results using different underestimating schemes.

$$\begin{aligned} \min \quad & f = -0.5 \sum_{i=1}^{20} \lambda_i (x_i - \beta_i)^2 \\ s.t. \quad & x \in P = \{x : Ax \leq b, x \geq 0\} \subset \Re^{20}. \end{aligned}$$

Five cases are defined for different values of (λ, β) as shown in Table 13.3. The global minima are also given in Table 13.3, while the values of the variables at the optimal solutions are given in Table 13.4.

	C_1	C_2	C_3	C_4	C_5
λ_i	1	1	20	1	i
β_i	2	-5	0	8	2
f^*	-394.7506	-884.7508	-8695.011	-754.7506	-4150.410

Table 13.3. Parameter and optimal objective function values for the five instances of the linearly constrained concave problems ($i = 1, \cdots, 20$).

Case	x_3	x_6	x_{11}	x_{13}	x_{15}	x_{16}	x_{18}	x_{20}
C_1	28.80	4.179	0	0	0.619	4.093	2.306	0
C_2	28.80	4.179	0	0	0.619	4.093	2.306	0
C_3	28.80	4.179	0	0	0.619	4.093	2.306	0
C_4	28.80	4.179	0	0	0.619	4.093	2.306	0
C_5	1.043	0	1.747	0.431	0	4.433	15.858	16.487

Table 13.4. Optimal solutions of the five instances of the linearly constrained concave problems – Variables not shown here are equal to zero.

Since this problem involves only linear and univariate concave terms, linear underestimators may be used to construct the lower bounding problem.

The computational requirements for the different strategies are summarized in Table 13.5. Results obtained with the GOP algorithm are also provided. A significant reduction of the number of iterations as well as the CPU time requirements is achieved when an appropriate branching procedure is used (Runs R3 and R4). Such a scheme repeatedly selects a few key variables for branching from the entire pool of 20 variables. When variable bounds updates are performed at every iteration (Runs R2 and R4), only a few variables bounds are actually optimized. This results in a significant decrease in the required number of iterations. However the overall CPU time increases by as much as a factor of 2. It was found that the construction of the lower bounding problem in each domain represented on the order of 1.0% of the overall computational expense when bound updates were performed at each iteration, and 0.1% when bound updates took place at the first iteration only.

	Iterations					CPU				
Case	R1	R2	R3	R4	GOP	R1	R2	R3	R4	GOP
C_1	137	72	42	26	27	13.2	23.2	4.8	9.4	0.68
C_2	116	71	38	23	4	11.6	22.9	4.5	8.2	3.57
C_3	123	77	41	26	11	11.3	24.2	4.6	8.5	10.91
C_4	115	79	38	24	5	11.9	25.2	4.4	8.3	5.07
C_5	544	221	134	63	229	51.0	67.0	12.0	19.9	177.04

R1 : Branching strategy 1; Bounds updates at first iteration,
R2 : Branching strategy 1; Bounds updates at every iteration,
R3 : Branching strategy 3; Bounds updates at first iteration,
R4 : Branching strategy 3; Bounds updates at every iteration,
GOP : GOP algorithm.

Table 13.5. Computational requirements for the five instances of the linearly constrained concave problems.

When the univariate concave terms are treated as general nonconvex terms and α values are calculated for each one of them, the computational requirements for convergence to the global optimum solution are almost unchanged, as can be seen in Table 13.6. Once again, the generation of the convex underestimators makes up about 1% of the computational cost.

Constrained Nonlinear Optimization Example. This is a small but nevertheless difficult nonconvex problem from Murtagh and Saunders (1983). The αBB algorithm is very well suited for this example since it combines bilinear, univariate concave and general nonconvex terms.

$$\min \quad (x_1 - 1)^2 + (x_1 - x_2)^2 + (x_2 - x_3)^3 + (x_3 - x_4)^4 + (x_4 - x_5)^4$$

s.t.

$$\begin{aligned} x_1 + x_2^2 + x_3^3 &= 3\sqrt{2} + 2 \\ x_2 - x_3^2 + x_4 &= 2\sqrt{2} - 2 \\ x_1 x_5 &= 2 \\ x_i &\in [-5, 5],\ i = 1, \cdots, 5. \end{aligned}$$

The global minimum as well as some local minima are presented in Table 13.7.

All the α calculation methods presented in Chapter 12 have been used to solve this problem to global optimality. Strategies 2, 3 or 4 were used for branching variable selection and bound updates were performed on a subset of the variables at every iteration for all runs. All branching strategies were tried for this problem and the most successful run is reported in Table 13.8.

The percentage of total time spent generating convex underestimators, t_U, varies from method to method. In the case of Method II.3, the increase in computational expense is due to the time spent solving semi-definite programs. For all other methods, a low percentage of the overall CPU time (about 2.5%) is needed to derive the lower bounding problem.

The best results are given by the E-matrix approach, the Hertz method and the scaled Gerschgorin approach with $d_i = (x_i^U - x_i^L)$. The minimization of the maximum separation distance approach converges with the smallest number of iterations but performs poorly in terms of CPU requirements. Branching strategy 3 is especially well-suited for this problem. If the variable bounds are updated only once at the beginning of the αBB run, the scaled Gerschgorin method requires 2026 iterations and 224.2 CPU seconds. The relatively high degree of nonlinearity of this example problem increases the dependence of the quality of the lower bounding problem on the variable bounds: while the α values were constant for the Haverly pooling problems and the linearly constrained concave problems, they are now computed through interval arithmetic. Variable bound updates should therefore be carried out preferentially on the most nonlinear variables.

For all the methods used, the quality of the lower bound increases sharply from iteration to iteration at the beginning of a run. After a few levels of the branch-and-bound tree have been explored, the rate of improvement tapers off. Figure 13.2 shows the progress of the lower bound for the first 100 iterations using Method II.2 and one can clearly see the dramatic amelioration in the quality of the underestimator.

The shape of the branch-and-bound tree also reflects the efficiency of the underestimating techniques used in the αBB algorithm: when using the Hertz method, for example, all nodes on the first five levels are visited. Fathoming becomes significant at subsequent levels as the lower bound approaches the optimum solution. Level 6 is 95% full, level 7 is 69% full, level 8 is 36% full and level 9 is only 25% full. Thereafter, the absolute number of visited nodes on each level actually decreases as the algorithm zeroes in on the region containing

the solution and time is spent closing the small gap between the lower and upper bounds.

13.3.2 Engineering Design Problems

Having established the importance of branching variable selection and variable bound updates, some results will now be presented for a set of optimization formulations for chemical process design problems. The first example corresponds to a simplified alkylation process and was also addressed in Chapter 9. The second problem is a small reactor network design problem addressed also in Chapter 9. The third example is a heat exchanger network design problem and the fourth example is a nonsharp separation network design problem which is also adressed via the GOP approach in Chapter 4.

Alkylation Process Design Problem. This example was studied in section 9.1.1 of Chapter 9 where the mathematical model is provided. We revisit it here with the only difference being that the objective function does not contain the constant term c_5. The maximum profit is \$1772.77 per day, and the optimal variable values are $x_1^* = 1698.18$, $x_2^* = 53.66$, $x_3^* = 3031.30$, $x_4^* = 90.11$, $x_5^* = 95.00$, $x_6^* = 10.50$, $x_7^* = 153.53$.

In this example, an update of all the variable bounds therefore necessitates the solution of 14 convex problems. Although bound tightening is likely to result in the construction of more accurate underestimators, its cost may outweigh the benefits. Two bounding strategies were used for this example: an update of all variable bounds at the onset of the algorithm, or an update of all bounds at each iteration of the branch-and-bound tree. The results are summarized in Table 13.9.

The results presented in the 'Single Up.' column were generated by updating all variable bounds only before the first iteration. The 'One Up./Iter' results were obtained by updating all bounds at every iteration. Although the second approach results in tighter underestimators, and hence a smaller number of iterations, the time requirements for each iteration are significantly larger than when no bounds updates are performed. Thus, the overall CPU requirements may be larger when all variable bounds are updated at each iteration. A comparison of the progress of the lower bound for each strategy as a function of iteration number and as a function of CPU time for the scaled Gerschgorin theorem method with $d_i = (x_i^U - x_i^L)$ (Method II.1), shown in Figure 13.3, illustrates this point. The percentage of overall computational effort dedicated to the construction of the convex lower bounding problem, t_U, is small for almost all methods. It is significantly larger for Method II.3 as it requires the solution of a semi-definite programming problem. t_U decreases when bound updates are performed at each iteration as a large amount of time is spent solving the bound updates problems. In this example, the scaled Gerschgorin approach (Method II.1) with $d_i = (x_i^U - x_i^L)$ gives the best results both in terms of number of iterations and CPU time.

Reactor Network Design. We studied this reactor network design problem in section 9.1.2 of Chapter 9 and its formulation is :

$$
\begin{aligned}
\min \quad & -x_4 \\
s.t. \quad & x_1 + k_1 x_1 x_5 = 1 \\
& x_2 - x_1 + k_2 x_2 x_6 = 0 \\
& x_3 + x_1 + k_3 x_3 x_5 = 1 \\
& x_4 - x_3 + x_2 - x_1 + k_4 x_4 x_6 = 0 \\
& x_5^{0.5} + x_6^{0.5} \leq 4 \\
& k_1 = 0.09755988 \\
& k_2 = 0.99 k_1 \\
& k_3 = 0.0391908 \\
& k_4 = 0.9 \\
& (0, 0, 0, 10^{-5}, 10^{-5}) \leq (x_1, x_2, x_3, x_4, x_5) \leq (1, 1, 1, 1, 16, 16).
\end{aligned}
$$

A number of close local minima exist, as shown in Table 13.10. If the univariate concave and bilinear terms are underestimated linearly, the αBB algorithm identifies the global minimum after 33 iterations and 5.5 CPU seconds. When the bilinear terms are treated as general nonconvex terms, the computational requirements increase to 157 iterations and 43.4 CPU seconds. In both situations, the generation of convex underestimators requires about 0.4% of the overall CPU time.

Heat Exchanger Network Design. This problem is taken from Floudas and Pardalos (1990), and was also studied in section 9.1.3 of Chapter 9 where its mathematical formulation is provided. The same optimal value of the objective function is obtained and it takes the αBB algorithm 244 iterations and 54.4 seconds to identify the global minimum when linear underestimators are used. With convex underestimators, 558 iterations and 141.4 CPU seconds are required.

Nonsharp Separation Network Synthesis. This problem is taken from Floudas and Aggarwal (1990), and was also studied in section 4.2.4 of Chapter 4. The superstructure for the separation is shown in Figure 13.4. The formulation of the cost minimization problem given in Floudas and Aggarwal (1990) has been slightly modified to reduce the number of variables. It now involves 22 variables and 16 equality constraints. It is expressed as

$$\min 0.9979 + 0.00432F_1 + 0.00432F_{13} + 0.01517F_2 + 0.01517F_9$$

s.t

$$
\begin{aligned}
F_1 + F_2 + F_3 + F_4 &= 300 \\
F_5 - F_6 - F_7 &= 0 \\
F_8 - F_9 - F_{10} - F_{11} &= 0 \\
F_{12} - F_{13} - F_{14} - F_{15} &= 0 \\
F_{16} - F_{17} - F_{18} &= 0 \\
F_{13}x_{A,12} - F_5 + 0.333F_1 &= 0 \\
F_{13}x_{B,12} - F_8x_{B,8} + 0.333F_1 &= 0 \\
-F_8x_{C,8} + 0.333F_1 &= 0 \\
-F_{12}x_{A,12} - 0.333F_2 &= 0 \\
F_9x_{B,8} - F_{12}x_{B,12} + 0.333F_2 &= 0 \\
F_9x_{C,8} - F_{16} + 0.333F_2 &= 0 \\
F_{14}x_{A,12} + 0.333F_3 + F_6 &= 30 \\
F_{10}x_{B,8} + F_{14}x_{B,12} + 0.333F_3 &= 50 \\
F_{10}x_{C,8} + 0.333F_3 + F_{17} &= 30 \\
x_{B,8} + x_{C,8} &= 1 \\
x_{A,12} + x_{B,12} &= 1 \\
0 \leq F_i &\leq 150 \\
0 \leq x_{j,i} &\leq 1
\end{aligned}
$$

where F_i denotes the total flowrate of the ith stream and $x_{j,i}$ denotes the fraction of component j in the ith stream.

The global optimal configuration is identified and the optimal concentrations are $x_{B,8} = x_{C,8} = 0.5$, $x_{A,12} = 0$ and $x_{B,12} = 1$. The optimal non-zero flowrates are $F_1 = 60$, $F_3 = 90$, $F_4 = 150$, $F_5 = F_7 = 20$, $F_8 = F_9 = 40$, $F_{12} = F_{14} = 20$ and $F_{16} = F_{18} = 20$.

The algorithm converged to the global solution in 15.2 CPU seconds and after 11 iterations, when the bilinear terms were underestimated linearly. When they were treated as general nonconvex terms, 49 iterations and 220.1 CPU seconds were required.

13.3.3 Stability Analysis of Nonlinear Systems

In this section, we revisit the six examples presented in the section of *Robust Stability Analysis* in Chapter 9.

Example 1. This problem involves 4 variables and 4 constraints, one of which is nonlinear. The system is found to be unstable with $k = 0.3417$. The results shown in Table 13.11 were obtained with the third branching strategy, using a single bound update. They correspond to the solution of the global optimality problem, in which wide bounds on k are used to try and identify the smallest possible value. If the actual value of the stability margin is of no interest, the

feasibility problem can be solved, in which the value of k is restricted to $[0, 1]$. In such a case, the algorithm can terminate when an upper bound on k has been found within this interval (the system is unstable), or when infeasibility has been asserted (the system is stable). For this example, an upper bound below 1 is obtained in less than 1 CPU sec.

Example 2. Contrary to the previous example, this 4 variable system is stable, with a stability margin of $k = 1.089$. In order to prove that the system is stable, one must show that the feasibility problem has no solution. Alternatively, the minimum stability margin can be identified using wider bounds on k. Clearly, global optimality runs are more time demanding as they span a larger solution space. The results are shown in Table 13.12.

Example 3. The global optimum solution of $k = 0.8175$ is identified using the fourth branching strategy and a single bound update. The fact that the system is unstable is identified in under 1 CPU sec. The results of the global optimality problem are shown in Table 13.13.

Example 4. With a minimum stability margin of 6.27, this system is stable. The perturbation frequency ω, which was eliminated from previous formulations, is kept in this example. At the global optimum solution, the value of ω is 0.986. The bounds used for this variable have a significant effect on the convergence rate of the algorithm. To illustrate this point, the feasibility problem was solved with $\omega \in [0, 1]$ and $\omega \in [0, 10]$. The fourth branching strategy was used and a single variable bound update was performed in the first node. The results are shown in Table 13.14.

Example 5. This formulation is used to determine the stability of Daimler Benz bus. It involves 4 variables and the exponent values range from 1 to 8. The minimum stability margin is $k = 1.2069$ and the problem is treated as a feasibility problem. The results are shown in Table 13.15.

Example 6. This final problem was developed to study the stability of the Fiat Dedra spark ignition engine. It involves 9 variables and is highly nonlinear. The solution of the stability problem shows that this system is stable and computational results, obtained with $\omega \in [0, 10]$, are reported in Table 13.16. The complexity of this problem also results in an increase in the time requirements for the underestimator generation, regardless of the method used.

Remark 13.3.1 The results for the set of generalized geometric programming and robust stability problems show that no single method for the generation of a diagonal shift matrix systematically outperforms the others.

Summary

In this chapter, we discussed the key issues in the implementation of the αBB global optimization approach and we presented extensive computational studies on medium size and certain large size problems that arise in engineering design and robust stability analysis. The computational results strongly suggest that the αBB approach can address these classes of problems to global optimality in a rather efficient way. These studies also demonstrate the flexibility of the available options and the effects of the bound updating and branching strategies. For additional computational studies of the presented implementation of the αBB global optimization approach to (i) generalized geometric programming problems, and (ii) large scale problems in batch process design under uncertainty, the reader is directed to Adjiman *et al.* (1998b) and Harding and Floudas (1997). Global optimization test problems for general twice differentiable constrained NLPs can be found in the recent book by Floudas *et al.* (1999).

Underes.	C1		C2		C3		C4		C5	
	N_i	CPU	N_i	CPU	N_i	CPU	N_i	CPU	N_i	CPU
Linear	42	4.8	38	4.5	41	4.6	38	4.4	134	12.0
Convex	42	4.2	38	3.8	41	3.8	38	3.9	134	9.3

Table 13.6. Computational results using different underestimators for the linearly constrained concave problems. Strategy R3 was used for all runs.

	obj	x_1	x_2	x_3	x_4	x_5
Global	0.0293	1.1166	1.2204	1.5378	1.9728	1.7911
Local 1	27.8719	-1.2731	2.4104	1.1949	-0.1542	-1.5710
Local 2	44.0221	-0.7034	2.6357	-0.0963	-1.7980	-2.8434
Local 3	52.9026	0.7280	-2.2452	0.7795	3.6813	2.7472
Local 4	64.8740	4.5695	-1.2522	0.4718	2.3032	4.3770

Table 13.7. Global and local minima of constrained nonlinear example of section 13.3.1.

Method		Iter.	CPU sec.	t_U (%)	Branch
Gerschgorin (I.1)		353	76.3	2.1	4
E-Matrix (I.2)	$E = 0$	337	70.7	2.8	3
E-Matrix (I.2)	$E = \text{diag}(\Delta H)$	339	72.9	2.3	2
Mori-Kokame (I.3)		611	137.5	2.3	3
Lower bounding Hessian (I.4)		346	72.8	2.4	3
Hertz (I.5)		334	72.2	2.9	3
Scaled Ger. (II.1)	$d_i = 1$	352	76.1	2.0	4
Scaled Ger. (II.1)	$d_i = (x_i^U - x_i^L)$	330	69.7	2.0	3
H-Matrix (II.2)		347	79.6	3.1	3
Min. Max. distance (II.3)		318	100.4	30.3	4

Table 13.8. Results for constrained nonlinear example of section 13.3.1. t_U denotes the percentage of total CPU time spent generating convex underestimators.

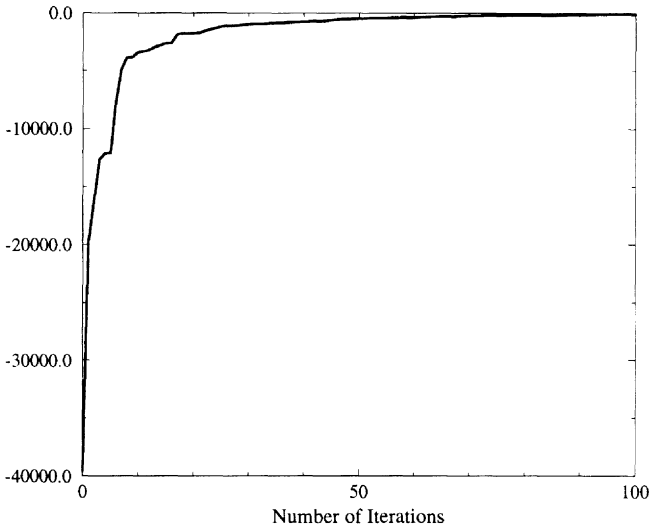


Figure 13.2. Progress of the lower bound for constrained nonlinear example of section 13.3.1.

Method		Single Up.			One Up./Iter		
		Iter.	CPU sec.	t_U (%)	Iter.	CPU sec.	t_U (%)
Gerschgorin (I.1)		74	37.5	0.5	31	41.6	0.0
E-Matrix (I.2)	$E = 0$	61	30.6	1.6	25	37.2	0.2
E-Matrix (I.2)	$E = \text{diag}(\Delta H)$	61	29.2	1.0	25	35.4	0.1
Mori-Kokame (I.3)		69	32.8	1.9	25	31.5	0.2
Lower bounding Hessian (I.4)		61	31.6	1.4	25	33.1	0.2
Hertz (I.5)		59	32.9	1.4	25	32.8	0.5
Scaled G. (II.1)	$d_i = 1$	56	24.9	0.3	30	36.5	0.3
Scaled G. (II.1)	$d_i = (x_i^U - x_i^L)$	38	13.6	1.7	17	19.9	0.5
H-Matrix (II.2)		62	32.7	0.6	25	34.5	0.3
Min. Max. distance (II.3)		54	21.8	16.7	23	30.4	5.0

Table 13.9. Alkylation Process Design Results for different α computation methods. t_U denotes the percentage of total CPU time spent generating convex underestimators.

	obj	x_1	x_2	x_3	x_4	x_5	x_6
Global	-0.38881	0.772	0.517	0.204	0.388	3.036	5.097
Local 1	-0.38808	1.0	0.393	0.0	0.388	10^{-5}	15.975
Local 2	-0.37461	0.391	0.391	0.375	0.375	15.975	10^{-5}

Table 13.10. Global and local minima of the small reactor network design problem.

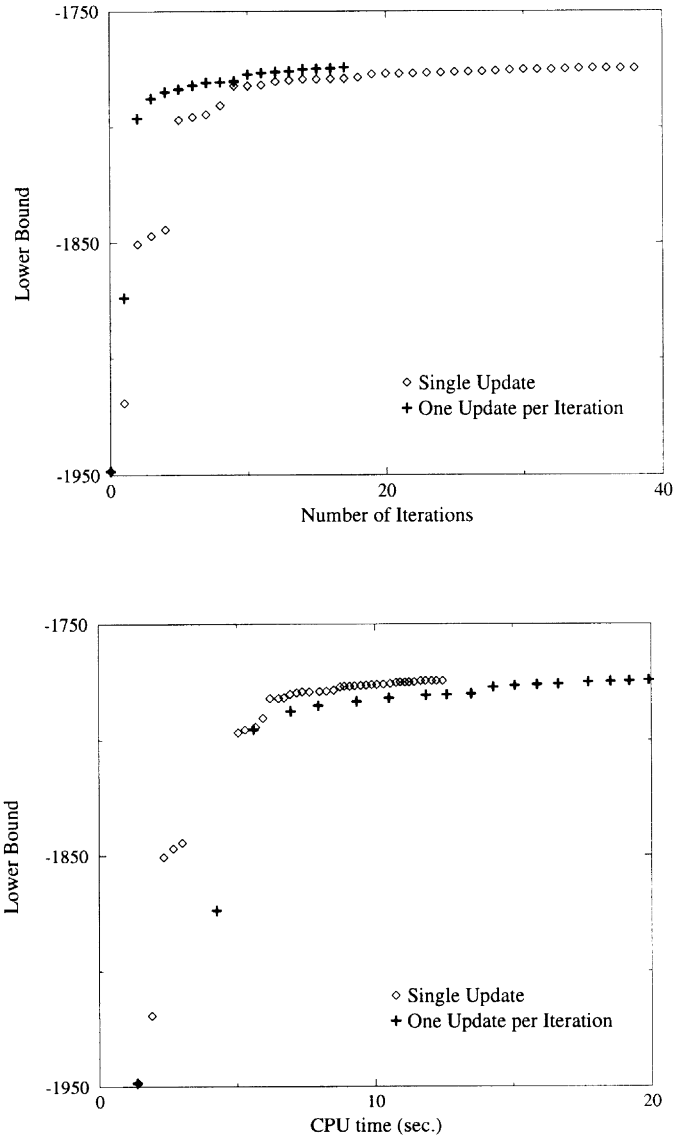


Figure 13.3. Lower bound vs. Iteration number or CPU time for the scaled Gerschgorin theorem approach in the Alkylation example.

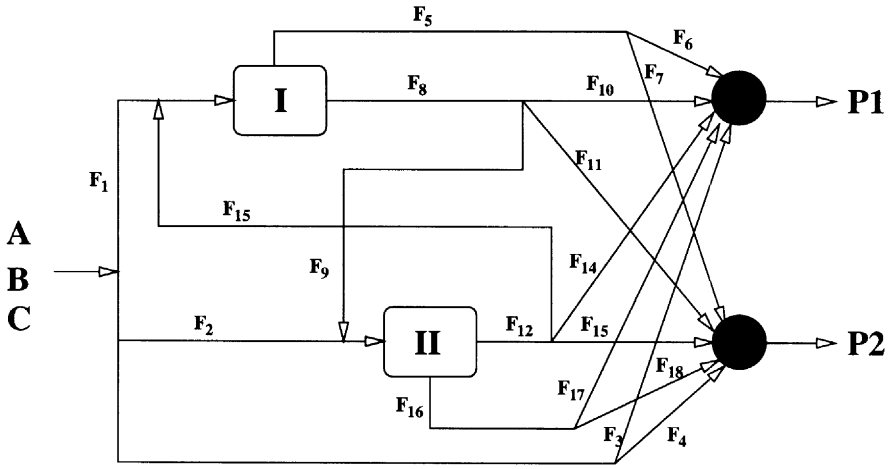


Figure 13.4. Superstructure for separation example

Method		Iter.	CPU sec.	t_U (%)
Gerschgorin (I.1)		24	1.4	26.5
E-Matrix (I.2)	$E = 0$	20	1.4	28.9
E-Matrix (I.2)	$E = \text{diag}(\Delta H)$	20	1.3	15.4
Mori-Kokame (I.3)		20	1.2	23.1
Lower bounding Hessian (I.4)		20	1.4	16.2
Hertz (I.5)		20	1.4	20.5
Scaled Ger. (II.1)	$d_i = 1$	22	1.6	11.1
Scaled Ger. (II.1)	$d_i = (x_i^U - x_i^L)$	21	1.3	15.2
H-Matrix (II.2)		21	1.4	19.3
Min. Max. distance (II.3)		18	8.6	85.0

Table 13.11. Results for Example 1 of Section 13.3.3. t_U denotes the percentage of total CPU time spent generating convex underestimators.

Method		Global Optimality			Feasibility		
		Iter.	CPU sec.	t_U (%)	Iter.	CPU sec.	t_U (%)
Gerschgorin (I.1)		82	3.0	3.5	31	1.1	7.9
E-Matrix (I.2)	$E = 0$	77	2.8	5.5	29	1.8	2.5
E-Matrix (I.2)	$E = \text{diag}(\Delta H)$	74	2.6	4.5	29	1.2	2.1
Mori-Kokame (I.3)		98	1.3	3.6	34	1.3	7.3
Lower bounding Hessian (I.4)		77	2.8	5.5	29	1.2	2.7
Hertz (I.5)		77	2.3	4.7	29	1.2	11.1
Scaled G. (II.1)	$d_i = 1$	43	1.5	1.7	13	0.8	5.6
Scaled G. (II.1)	$d_i = (x_i^U - x_i^L)$	35	1.2	2.0	13	0.7	5.6
H-Matrix (II.2)		76	3.3	5.2	29	1.6	2.8
Min. Max. distance (II.3)		34	5.1	86.1	11	2.9	75.1

Table 13.12. Results for Example 2 of Section 13.3.3. t_U denotes the percentage of total CPU time spent generating convex underestimators.

Method		Iter.	CPU sec.	t_U (%)
Gerschgorin (I.1)		20	1.0	4.3
E-Matrix (I.2)	$E = 0$	18	0.9	4.2
E-Matrix (I.2)	$E = \text{diag}(\Delta H)$	17	0.9	3.7
Mori-Kokame (I.3)		17	0.8	4.2
Lower bounding Hessian (I.4)		20	1.0	4.0
Hertz (I.5)		18	0.8	4.2
Scaled Ger. (II.1)	$d_i = 1$	17	0.8	4.2
Scaled Ger. (II.1)	$d_i = (x_i^U - x_i^L)$	12	0.6	5.3
H-Matrix (II.2)		17	0.9	4.4
Min. Max. distance (II.3)		26	2.1	37.6

Table 13.13. Results for Example 3 of Section 13.3.3. t_U denotes the percentage of total CPU time spent generating convex underestimators.

Method		$\omega \in [0,1]$ Iter.	CPU sec.	t_U (%)	$\omega \in [0,10]$ Iter.	CPU sec.	t_U (%)
Gerschgorin (I.1)		11	1.7	22.6	32	4.1	18.4
E-Matrix (I.2)	$E = 0$	9	1.2	22.7	29	3.2	13.8
E-Matrix (I.2)	$E = \mathrm{diag}(\Delta H)$	9	1.3	20.5	29	3.2	23.9
Mori-Kokame (I.3)		12	1.7	24.5	34	4.2	23.7
Lower bounding Hessian (I.4)		5	1.0	30.0	25	2.9	21.0
Hertz (I.5)		8	1.7	41.8	29	4.6	46.7
Scaled G. (II.1)	$d_i = 1$	1	0.6	18.8	8	1.3	21.2
Scaled G. (II.1)	$d_i = (x_i^U - x_i^L)$	1	0.4	14.2	5	0.8	21.2
H-Matrix (II.2)		9	1.3	16.1	28	3.4	15.6
Min. Max. distance (II.3)		1	2.6	91.2	8	16.5	58.4

Table 13.14. Results for Example 4 of Section 13.3.3. t_U denotes the percentage of total CPU time spent generating convex underestimators.

Method		$\omega \in [0,1]$ Iter.	CPU sec.	t_U (%)	$\omega \in [0,10]$ Iter.	CPU sec.	t_U (%)
Gerschgorin (I.1)		23	3.8	20.8	2997	784	18.9
E-Matrix (I.2)	$E = 0$	18	2.9	23.1	2559	392	22.6
E-Matrix (I.2)	$E = \text{diag}(\Delta H)$	20	3.4	26.3	2563	351	20.4
Mori-Kokame (I.3)		35	5.5	21.7	5328	1874	14.4
Lower bounding Hessian (I.4)		18	2.7	21.1	2695	111	22.1
Hertz (I.5)		16	2.7	24.3	2403	328	22.8
Scaled G. (II.1)	$d_i = 1$	18	2.5	22.4	1013	255	22.4
Scaled G. (II.1)	$d_i = (x_i^U - x_i^L)$	7	1.1	26.7	821	107	13.9
H-Matrix (II.2)		18	3.1	14.0	2457	385	12.4
Min. Max. distance (II.3)		14	5.4	69.8	1011	514	72.5

Table 13.15. Results for Example 5 of Section 13.3.3. t_U denotes the percentage of total CPU time spent generating convex underestimators.

Method		$\omega \in [0,10]$ Iter.	CPU sec.	t_U (%)
Gerschgorin (I.1)		148	42.3	63.4
E-Matrix (I.2)	$E = 0$	140	44.4	64.4
E-Matrix (I.2)	$E = \text{diag}(\Delta H)$	186	52.0	62.9
Mori-Kokame (I.3)		261	74.5	65.7
Lower bounding Hessian (I.4)		130	38.6	60.9
Hertz (I.5)		135	76.9	77.1
Scaled Ger. (II.1)	$d_i = 1$	163	48.9	63.7
Scaled Ger. (II.1)	$d_i = (x_i^U - x_i^L)$	3944	1129.2	64.0
H-Matrix (II.2)		143	45.1	71.3
Min. Max. distance (II.3)		246	1528.4	96.5

Table 13.16. Results for Example 6 of Section 13.3.3. t_U denotes the percentage of total CPU time spent generating convex underestimators.

14 GLOBAL OPTIMIZATION IN MICROCLUSTERS

In this chapter, we introduce the reader to the fundamental developments of global optimization for the structure prediction of clusters. Section 14.1 presents an introduction to the research area of microclusters and provides some background information. Section 14.2 discusses the problem definition. Section 14.3 presents the novel difference of convex functions (DC) transformation which is based on eigenvalue analysis for a variety of potential energy models, and which results in explicit ways of defining the parameter α. Section 14.4 presents the mathematical model, along with its modifications, that satisfy the convergence conditions of the GOP algorithm. Section 14.5 discusses the detailed algorithmic procedure for determining the global minimum structure of microclusters. Section 14.6 presents computational results on small microclusters. Section 14.7 introduces a relaxation of the global optimization approach that allows for the computational studies of larger Lennard-Jones and Morse clusters. Finally, section 14.8 discusses the results with reference to magic numbers and interatomic spacing. The material presented in the chapter is based on the work of Maranas and Floudas (1992), (1993).

14.1 INTRODUCTION

Microclusters (or clusters) belong to an intriguing state of matter that encompasses different entities such as micro-crystallites, microdroplets, colloid nuclei, aerosol particles, and grains. Hoare (1979) defined microclusters as aggregates

C. A. Floudas, *Deterministic Global Optimization*
© Springer Science+Business Media Dordrecht 2000

of atoms, ions, or molecules, sufficiently small so that a significant proportion of these units is present on their surfaces. They correspond to systems that are neither single entities nor continua composed of an infinite number of units, but lie somewhere in between, bridging the gap between single atoms or molecules and bulk matter. Typically, microclusters consist of two to several hundred atoms and are important for solid-state physics, chemistry, and material science. The importance of microclusters ranges from catalysis to astrophysics and crosses the boundaries of subjects such as nucleation, crystal growth and surface physics.

The study of the topography of the potential energy function of a microcluster in the internal configurational space is an important problem in this area (Hoare, 1979; Hoare and McInnes, 1983). Even when simplifying assumptions about the interaction energy, the minimization of the total potential energy is very difficult to solve because it corresponds to a nonconvex optimization problem involving numerous local minima. Hoare (1979) claimed that the number of local minima of an n–atom microcluster grows as $exp(n^2)$. Wille and Vennik (1985) showed that the complexity of determining the global minimum energy of a cluster of particles interacting via two-body forces belongs to the **NP**-class, that is, there is no known algorithm that can solve this problem in non-exponential time (Garey and Johnson, 1979). A geometrical, possibly topological proof that a local minimum is both unique and global is not likely to be found because there still exist unsolved problems in the theory of sphere packings where difficulties are considerably less acute than the minimization problem at hand (Coxeter, 1961; Fejes-Toth, 1954; Conway and Sloane, 1988).

Faced with these difficulties, current methods use physical intuition, approximation procedures, mimicking of physical phenomena, random searches, lattice optimization/relaxation, or local optimization approaches. Hoare, in a series of papers (Hoare and Pal (1971b) and (Hoare and McInnes (1983)), proposed a method of finding minima of the total potential function of an $N \leq 66$ particle Lennard-Jones cluster based on a *growth scheme.* Using this method Hoare generated a large number of local minima for structures ranging from 5 to 66 particles. However, no claim for complete enumeration of all local minima, and thus detection of the global minimum, can be made. Wille (1986) reported that solutions of low–symmetry are not likely to be found with this method.

Piela *et al.* (1989) proposed a method based on the idea of smoothly deforming the potential energy hypersurface, in such a way as to make shallow potential wells disappear gradually, while the deeper ones grow at their expense. The hypersurface is deformed using the diffusion equation, with the original shape of the hypersurface representing the initial concentration distribution. The main advantage of this method is that you do not have to explore the myriads of local optima, nor do you have to know their position beforehand. However, the approach depends on the conjecture that shallow potential wells disappear faster than deeper ones.

Simulation studies are the most widespread method in the study of clusters. There are three basic simulation methods that have been used in cluster research: molecular dynamics (MD), Monte Carlo simulations and statics.

Simulation calculations using the (MD) technique were carried out by Jellinek *et al.* (1986) for studying the solid liquid changes in a 13–particle argon cluster. A systematic molecular dynamics simulation study of microclusters of 13–147 atoms was performed by Honeycutt and Andersen (1987) using the Lennard–Jones potential at constant temperature. Polymeropoulos and Brickmann (1983) employed an (MD) technique using two–body and three-body interactions calculating the stabilities of microclusters of various sizes and analyzing the effect of three-body interactions on the stability of the microcluster. Polymeropoulos and Brickmann (1985) also analyzed ionized rare–gas clusters containing up to 26 atoms. Garzon *et al.* (1989b) analyzed the effect of impurities on the structure and on the liquid-solid transition of Lennard–Jones and Van der Waals crystals, respectively. Erkoc and Katircioglu (1988) investigated the energetics and the structural stability of gold microclusters containing 3–7 atoms. The potential energy function used in the calculations included two as well as three–body interactions represented by Lennard–Jones and Axilrod–Teller functions respectively.

Monte Carlo techniques, employed in microcluster simulations, are in general based on the Metropolis procedure (Metropolis *et al.*, 1953). Starting from an initial random configuration, atoms are randomly displaced according to the Maxwell–Boltzmann distribution. After a sufficient number of Monte Carlo steps for sampling the phase space have been taken, the desired quantities are calculated as ensemble averages from position–dependent quantities estimated in every step. With this method any equilibrium quantity can be calculated as a function of temperature which is introduced via the Maxwell–Boltzmann factor. Bohmer and Peyerimhoff (1989) analyzed the stability and structure of single–charged argon clusters of 3 to 27 atoms using Monte Carlo calculations in the temperature range between 10 K and 40 K. Freeman and Doll (1984) performed a quantum Monte Carlo study of the thermodynamic properties, homogeneous nucleation, and "magic numbers" distribution in argon clusters.

The static method is based on a minimization technique to find the configuration of a cluster corresponding to the nearest energy local minimum. It is a temperature–independent approach and can be regarded as the $T = 0\ K$ case.

The simulated annealing method has been widely used either alone, or in conjunction with some other method(s). The motivation of this method stems from the fact that each time a crystal is grown or a compound is synthesized, nature somehow seems to solve an energy minimization problem by what could be called an analog computation. It was this observation that led Kirkpatrick *et al.* (1983) to develop the simulated annealing method for discrete combinatorial optimization problems. The simulated annealing procedure does not involve any deterministic mathematical guarantees for convergence to the global minimum. If the catchment region of the global minimum is narrow, it has been reported that this method is likely to miss it. A large number

of researchers have used this method in an attempt to locate the the global minimum of various potential energy functions. Wille (1986, 1987) addressed the potential minimization problem for up to 25 particles, interacting under two-body Lennard-Jones forces and found two new minima for $N = 24$ that were better than those reported by Hoare (1979). Ballone and Milani (1990) using a semi-empirical many-body potential, identified the ground–geometries of carbon clusters in the range $50 \leq M \leq 72$, and found that all the structures of low energies are hollow spheres with nearly graphitic atomic arrangement. Hohl *et al.* (1988) applied the same methodology to phosphorus clusters P_2 to P_8. Navon *et al.* (1990) used a combined simulated annealing and a quasi–Newton–like conjugate–gradient method for determining the structure of mixed argon–xenon clusters interacting with two–body Lennard–Jones forces. Vlachos *et al.* (1992a, 1992b) studied the binding energy of Ni Lennard–Jones clusters using the simulated annealing method in a canonical ensemble Monte Carlo technique. Shalloway (1992) presented a deterministic method for annealing the objective function by tracing the evolution of a multiple–Gaussian–packet approximation, while employing notions from renormalization group theory.

Farges *et al.* (1983) proposed that the most energetically favored microclusters in the range $20 \leq N \leq 50$ are the ones that involve interpenetrating icosahedra (polyicosahedra) or (PIC). For $N \leq 55$ a double icosahedral (DIC) growth scheme was introduced (Farges *et al.*, 1985) and for $55 \leq N \leq 147$ (Farges *et al.*, 1986) a third layer icosahedral structure using two different surface arrangements was presented. Using these notions Northby (1987) derived low energy configurations for LJ microclusters in the range $13 \leq N \leq 147$ based on a lattice optimization/relaxation algorithm. First, a heuristic procedure is employed for finding a set of lattice local minimizers assuming icosahedral IC or face–centered FC arrangements. Then, the currently *best* lattice minimizers are relaxed by using a local optimization algorithm. Xue (1994b) improved on Northby's method by reducing the algorithmic complexity. Furthermore, by relaxing *every* lattice local minimizer a number of good configurations were found in the range $13 \leq N \leq 147$. It appears that the best local lattice does not always relax to the structure involving the lowest total Lennard–Jones potential energy. By employing a parallel version of a two–level simulated annealing algorithm Xue (1994a) found energy minima for cluster sizes as large as $N = 100,000$. In the next section, the specific problem addressed in this work is formally defined and the adopted assumptions are outlined.

The excellent review paper by Berry (1993) provides a comprehensive account of all the approaches proposed up to that point, while the Cambridge Cluster Database (http://brian.ch.cam.ac.uk/CCD.html) represents a state of the art reference for the developments on Lennard-Jones clusters, C_{60} clusters, TIP4P water clusters, Morse clusters, and Sutton-Chen clusters, among others. Also, the recent book on test problems for local and global optimization by Floudas *et al.* (1999) describes mathematical formulations and results on clusters that include Lennard-Jones, Morse, Tersoff, Brenner, and Bolding-Andersen potentials.

14.2 PROBLEM DEFINITION

The problem which is to be addressed in this chapter is stated as follows:

Given N particles interacting with two-body central forces, find their configuration(s) in the three-dimensional Euclidean space possessing the global minimum total potential energy.

The employed simplifications in this work are the following:

(i) Many–body and angle dependent interactions are not taken into account.

(ii) Conformational entropy is not taken into consideration.

(iii) All particles are considered to be spherical and of the same size.

Assumption (i) restricts the applicability of the global optimization approach as developed to systems of particles interacting with two–body central forces. Although this model is certainly quite restrictive for some real systems because it ignores angle dependent force contributions and many–body force terms, it provides a valuable prototype for the study of microcluster mechanics. Note that the proposed approach can conceptually be extended to systems of particles interacting with many–body forces if an expression for the total potential energy as a function of the Euclidean distances exists. Assumption (ii) implies that the static microcluster configuration stability problem is addressed in this work. Assumption (iii) implies that all particles are identical and are of the same size. It has been introduced for the sake of simplicity and can be relaxed easily.

Under the aforementioned assumptions the potential energy of a N–particle cluster can be written as:

$$\begin{aligned} V &= \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} v(r_{ij}) \\ \text{where} \quad r_{ij} &= \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2} \\ \text{and} \quad & x_1 = y_1 = z_1 = y_2 = z_2 = z_3 = 0. \end{aligned}$$

Here V is the total potential energy of the microcluster as the summation of all two-body interaction terms, $v(r_{ij})$ is the potential energy term corresponding to the interaction of particle i with particle j, and r_{ij} is the Euclidean distance between i and j. Note that in the double summation, j spans from $i+1$ to N so that we avoid double counting pair interactions and the interaction of a particle with itself. Furthermore, by specifying $x_1 = y_1 = z_1 = 0$, we fix the first particle at $(0,0,0)$ eliminating all three translational degrees of freedom for the microcluster. By further imposing $y_2 = z_2 = z_3 = 0$, we eliminate the rotational degrees of freedom as well.

Many different formulas for $v(r)$ have been introduced. In order for $v(r)$ to have satisfactory behavior in the entire configurational space, it must comply with the following requirements as set by Hoare (1979) :

(i) $v(r) \to 0^-$ as $r \to \infty$ *(No interaction at infinite separation)*,

(ii) $v(r) \to \infty$ for $r < r_{min}$ and $r_{min} \geq 0$ *(Rigid core)*,

(iii) $v'(r_o) = 0$, $v(r_o)'' > 0$ for a unique r_o *(Single stable state)* .

Pair potentials that have been used in cluster studies include the following (Hoare, 1979):

(i) $v(r) = (n-m)^{-1}[nr^{-m} - mr^{-n}]$ (Mie),

(ii) $v(r) = 4\epsilon\left\{\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6}\right\}$ (Lennard–Jones),

(iii) $v(r) = [1 - e^{a(1-r)}]^2 - 1$ (Morse),

(iv) $v(r) = Ae^{-ar^2} - Be^{-br^2}$ (Gaussian),

(v) $v(r) = z^{\alpha}z^{\beta}/r + Ae^{-r/\rho}$ (Born–Meyer).

A necessary but not sufficient condition for local optimality is that all first order derivatives must be zero. Posing the problem on the x_i, y_i, z_i Cartesian coordinates we can write for any local minimum point:

$$\sum_{j=1}^{N} \frac{v'(r_{ij})}{r_{ij}}(x_i - x_j) = 0, \quad \forall i = 2, \ldots, N$$

$$\sum_{j=1}^{N} \frac{v'(r_{ij})}{r_{ij}}(y_i - y_j) = 0, \quad \forall i = 3, \ldots, N$$

$$\sum_{j=1}^{N} \frac{v'(r_{ij})}{r_{ij}}(z_i - z_j) = 0, \quad \forall i = 4, \ldots, N.$$

Note that $v'(r_{ij})$ is the first order derivative of $v(r_{ij})$ with respect to the Euclidean distance r_{ij}. The points that conform with the above conditions are called stationary points, and their number increases more than exponentially with N (Hoare and McInnes, 1983). A portion of the stationary points are local minima, however, others may be saddle points or local maxima. The second condition required in order to guarantee local minimality is that the Hessian matrix H of V with respect to the Cartesian coordinates x_i, y_i, z_i, $i = 1, \ldots, N$ must be positive definite at the point in question. The Hessian matrix contains the second order derivatives with respect to x_i, y_i, z_i of the total potential energy function. An equivalent condition to a positive definite Hessian matrix is requiring that all eigenvalues λ, are positive at the point in question. A physical interpretation of the eigenvalues of H is that at any point they give the principal radii of curvature, and the corresponding eigenvectors provide the principal axes and the set of normal coordinates in terms of which the Hessian is diagonal.

N	# Local Minima
6	2
7	4
8	8
9	18
10	57
11	145
12	366
13	>988

Table 14.1. Number of local minimum configurations of LJ microclusters $6 \leq N \leq 13$

Since all particles are assumed to be identical, we must consider the multiplicity of configurations which differ only by permutations of the particles. In fact, for a N–particle cluster there are $(N-3)!$ identical structures. These structures, which differ only in labeling of particles, are equivalent under proper symmetry operations in three–space, and correspond to enantiomorphic configurations (Hoare, 1979) that are called *geometrically equivalent.*

Remark 14.2.1 The main difficulty in solving this problem arises from the fact that the objective is a nonconvex function of many variables. Any path following technique is likely to find only a local minimum at best. Hoare and McInnes (1983) attempted the enumeration of many local minima of Lennard–Jones microclusters. The results shown on Table 14.1 clearly indicate that the number of known local minima increases very rapidly with N.

The approach introduced by Maranas and Floudas (1992), (1993) circumvents the existence of myriads of local minima in the potential energy hypersurface by requiring the solution of a series of only convex minimization problems involving a single minimum.

14.3 NOVEL D.C. TRANSFORMATION VIA EIGENVALUE ANALYSIS

The main difficulty in solving the minimization problem, as presented in the previous section, arises from the presence of nonconvexities in the objective function. In fact, we will show that even the expression for a single pair interaction potential is a nonconvex function. Let $v(r)$ be the general expression for the pair interaction potential between particle (0) and particle (1). If we now pose the problem in the Cartesian coordinate space, $v(r)$ becomes a function

of six variables x_0, y_0, z_0, x, y, z. However, particle (0) can be arbitrarily fixed at (x_0, y_0, z_0) and thus $v(r)$ is essentially a function of only three variables. A necessary and sufficient condition for convexity is that all the eigenvalues of the Hessian matrix of $v(r)$ must be positive. The first and second order derivatives of v with respect to x, y, z are the following:

$$\begin{aligned} h_x &= \frac{v'(r)}{r}(x - x_0) \\ h_y &= \frac{v'(r)}{r}(y - y_0) \\ h_z &= \frac{v'(r)}{r}(z - z_0) \end{aligned}$$

$$\begin{aligned} h_{xx} &= \left(v''(r) - \frac{v'(r)}{r}\right)\frac{(x - x_0)^2}{r^2} + \frac{v'(r)}{r} \\ h_{yy} &= \left(v''(r) - \frac{v'(r)}{r}\right)\frac{(y - y_0)^2}{r^2} + \frac{v'(r)}{r} \\ h_{zz} &= \left(v''(r) - \frac{v'(r)}{r}\right)\frac{(z - z_0)^2}{r^2} + \frac{v'(r)}{r} \\ h_{xy} &= \left(v''(r) - \frac{v'(r)}{r}\right)\frac{(x - x_0)(y - y_0)}{r^2} \\ h_{xz} &= \left(v''(r) - \frac{v'(r)}{r}\right)\frac{(x - x_0)(z - z_0)}{r^2} \\ h_{yz} &= \left(v''(r) - \frac{v'(r)}{r}\right)\frac{(y - y_0)(z - z_0)}{r^2}. \end{aligned}$$

Here $v'(r), v''(r)$ correspond to the first and second derivative of $v(r)$ with respect to the Euclidean distance r. The three eigenvalues $\lambda_1, \lambda_2, \lambda_3$ can then be calculated from the following equation.

$$\det \begin{pmatrix} h_{xx} - \lambda & h_{xy} & h_{xz} \\ h_{xy} & h_{yy} - \lambda & h_{yz} \\ h_{xz} & h_{yz} & h_{zz} - \lambda \end{pmatrix} = 0.$$

By expanding the determinant we obtain:

$$\begin{aligned} &- \lambda^3 \\ &+ \lambda^2(h_{xx} + h_{yy} + h_{zz}) \\ &+ \lambda(h_{xy}^2 + h_{xz}^2 + h_{yz}^2 - h_{xx}h_{yy} - h_{xx}h_{zz} - h_{yy}h_{zz}) \\ &+ (h_{xx}h_{yy}h_{zz} - h_{xy}^2 h_{zz} - h_{xz}^2 h_{yy} - h_{yz}^2 h_{xx} + 2h_{xy}h_{xz}h_{yz}) = 0. \end{aligned}$$

After substituting the expressions for $h_{xx}, h_{yy}, h_{zz}, h_{xy}, h_{xz}, h_{yz}$ the above equation becomes:

$$-\lambda^3 + \lambda^2\left(v''(r) + 2\frac{v'(r)}{r}\right) - \lambda\frac{v'(r)}{r}\left(v''(r) + 2\frac{v'(r)}{r}\right) + v''(r)\left(\frac{v'(r)}{r}\right)^2 = 0.$$

Surprisingly by factoring the above equation we obtain the following simple formula,

$$\left(\lambda - \frac{v'(r)}{r}\right)^2 (\lambda - v''(r)) = 0$$

which yields the following expressions for the three eigenvalues:

$$\begin{aligned} \lambda_{1,2} &= \frac{v'(r)}{r} \\ \lambda_3 &= v''(r). \end{aligned}$$

As a result, we have that :

$v(r)$ is convex in the Cartesian coordinate space x, y, z if and only if $v'(r)$ and $v''(r)$ are nonnegative for every x, y, z.

This result is valid for any type of potential describing the interaction of two particles with central forces since no assumptions for the functionality between r and $v(r)$ have been made. Furthermore, convexity of $v(r)$ in the r space is not a sufficient condition for convexity in the x, y, z Cartesian coordinate space. If $v(r)$ is convex in r then $v''(r)$ will be indeed positive, however $v'(r)$ might not be. Because the expression for $v(r)$ must comply with the requirements set by Hoare (1979), $v'(r)$ is negative for $r \leq r_0$ where r_0 is the value of r at which $v'(r)$ becomes equal to zero,

$$r_0 = \{r : v'(r) = 0\}.$$

Furthermore, $v''(r)$ is negative for $r \geq r_1$. where r_1 is the value of r at which $v''(r)$ becomes equal to zero,

$$r_1 = \{r : v''(r) = 0\}.$$

This implies that $v(r)$ stays convex only within the narrow interval (r_0, r_1). For different pair potential models, Table 14.2 provides the expressions for the eigenvalues $\lambda_{1,2}, \lambda_3$ and Table 14.3 the analytical formulas (if they exist) for r_0 and r_1. Figure 14.1 illustrates the scaled Lennard–Jones and Morse potentials.

The very important property of convexity of $v(r)$ can be maintained by augmenting $v(r)$ with the addition of a "strongly" convex term. This term may have the following form:

$$\alpha(x_0^2 + y_0^2 + z_0^2 + x^2 + y^2 + z^2).$$

Note that α is an arbitrarily large positive number. The effect of this extra term in the Hessian matrix is to add the term 2α to all the diagonal elements of the matrix which, given a sufficiently large α, forces all the eigenvalues to become non-negative by overpowering the original nonconvexity characteristics of $v(r)$. Let $\phi(r)$ be the summation of $v(r)$ and the extra term

Potential	$v(r)$	$\lambda_{1,2}$	λ_3
Coulomb	$\frac{z^\alpha z^\beta}{r}$	$-\frac{z^\alpha z^\beta}{r^3}$	$\frac{2z^\alpha z^\beta}{r^3}$
Scaled LJ	$\frac{1}{r^{12}} - \frac{2}{r^6}$	$12\left(\frac{1}{r^8} - \frac{1}{r^{14}}\right)$	$12\left(\frac{13}{r^{14}} - \frac{7}{r^8}\right)$
LJ	$4\epsilon\left\{\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^6\right\}$	$\frac{24\epsilon}{\sigma^2}\left[\left(\frac{\sigma}{r}\right)^8 - \left(\frac{\sigma}{r}\right)^{14}\right]$	$\frac{24\epsilon}{\sigma^2}\left[26\left(\frac{\sigma}{r}\right)^{14} - 7\left(\frac{\sigma}{r}\right)^8\right]$
Mie	$-\frac{1}{n-m}\left[\frac{n}{r^m} - \frac{m}{r^n}\right]$	$-\frac{nm}{n-m}\left[\frac{1}{r^{n+2}} - \frac{1}{r^{m+2}}\right]$	$-\frac{nm}{n-m}\left[\frac{(m+1)}{r^{m+2}} - \frac{(n+1)}{r^{n+2}}\right]$
Morse	$\left[1 - e^{a(1-r)}\right]^2 - 1$	$\frac{2a}{r}e^{a(1-r)}\left[1 - e^{a(1-r)}\right]$	$2a^2e^{a(1-r)}\left[2e^{a(1-r)} - 1\right]$
Gaussian	$Ae^{-ar^2} - Be^{-br^2}$	$2\left(bBe^{-br^2} - aAe^{-ar^2}\right)$	$4r^2\left(a^2Ae^{-ar^2} - b^2Be^{-br^2}\right)$
Born–Meyer	$\frac{z^\alpha z^\beta}{r} + Ae^{-r/\rho}$	$\frac{-z^\alpha z^\beta}{r^3} - \frac{A}{\rho r}e^{-r/\rho}$	$\frac{2z^\alpha z^\beta}{r^3} + \frac{A}{\rho^2}e^{-r/\rho}$

Table 14.2. Eigenvalues $\lambda_{1,2}, \lambda_3$ for different potential models

Potential	$v(r)$	r_0	r_1
Scaled LJ	$\frac{1}{r^{12}} - \frac{2}{r^6}$	1	$\left(\frac{13}{7}\right)^{\frac{1}{6}} \approx 1.108683$
LJ	$4\epsilon\left\{\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^6\right\}$	$2^{\frac{1}{6}}\sigma$	$\left(\frac{26}{7}\right)^{\frac{1}{6}}\sigma$
Mie	$-\frac{1}{n-m}\left[\frac{n}{r^m} - \frac{m}{r^n}\right]$	1	$\left(\frac{n+1}{m+1}\right)^{\frac{1}{n-m}}$
Morse	$\left[1 - e^{a(1-r)}\right]^2 - 1$	1	$\frac{\ln 2}{a} + 1$

Table 14.3. Convex interval (r_0, r_1) for different potential models

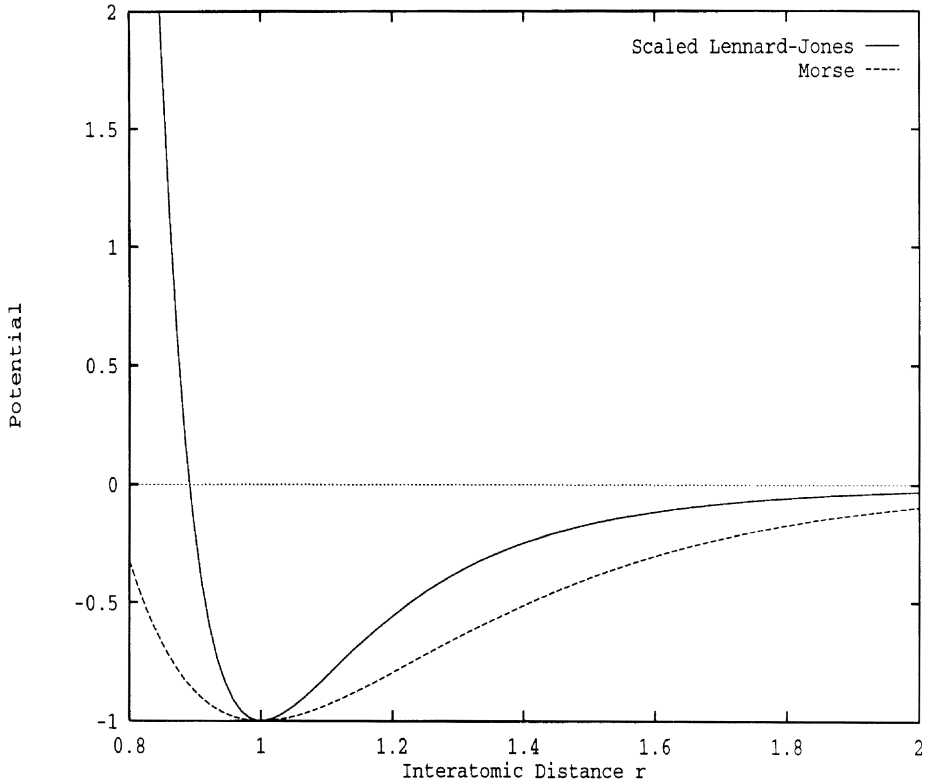


Figure 14.1. Plots of scaled Lennard–Jones and Morse potentials

$\alpha\left(x_0^2+y_0^2+z_0^2+x^2+y^2+z^2\right)$:

$$\phi(r) \quad = \quad v(r)+\alpha\left(x_0^2+y_0^2+z_0^2+x^2+y^2+z^2\right).$$

It is quite straightforward to show that the eigenvalues of the Hessian which corresponds to $\phi(r)$ are:

$$\begin{aligned}\lambda_{1,2} &= \frac{v'(r)}{r}+2\alpha \\ \lambda_3 &= v''(r)+2\alpha.\end{aligned}$$

Since we want all the eigenvalues to be nonnegative, we have the following inequality constraints for α.

$$\begin{aligned} \alpha &\geq 0 \\ \alpha &\geq -\frac{v'(r)}{2r} \\ \alpha &\geq -\frac{v''(r)}{2}. \end{aligned}$$

The minimum value of parameter α that will maintain all eigenvalues of $\phi(r)$ nonnegative can then be derived as the solution of the following optimization problem in a single variable r.

$$\alpha = \max_r \begin{cases} -\frac{v'(r)}{2r} & \text{if } r_{min} \leq r \leq r_0 \\ 0 & \text{if } r_0 \leq r \leq r_1 \\ -\frac{v''(r)}{2} & \text{if } r_1 \leq r \leq r_{max} \end{cases}.$$

Here r_{min} and r_{max} are the minimum and maximum Euclidean distances, respectively. For the scaled Lennard–Jones potential,

$$v_{LJ}(r) = \frac{1}{r^{12}} - \frac{2}{r^6}$$

this optimization problem takes the following form:

$$\alpha = \max_r \begin{cases} 6\left(\frac{1}{r^{14}} - \frac{1}{r^8}\right) & \text{if } r_{min} \leq r \leq 1 \\ 0 & \text{if } 1 \leq r \leq 1.109 \\ 6\left(\frac{7}{r^8} - \frac{13}{r^{14}}\right) & \text{if } 1.109 \leq r \leq r_{max} \end{cases}.$$

The pictorial solution of this problem for the scaled Lennard–Jones potential is illustrated in Figure 14.2. Note that at the global minimum all interparticle distances are never significantly less than one, because otherwise the corresponding pair potential assumes a large positive value which cannot occur at the global minimum.

Remark 14.3.1 It is very important for the presented (DC) transformation that all the eigenvalues of the Hessian matrix do not diverge to $-\infty$. This is achieved only if all Euclidean distances between the particles are strictly greater than zero at the global minimum. In addition, the larger r_{min} becomes, the tighter the lower bounding of the objective function.

This motivates not only the proof of existence of an $r_{min} > 0$, but also the identification of the tightest possible one. Maranas and Floudas (1992) proposed rigorous lower bounds on the minimum interparticle distances for different cluster sizes, as the following theorem suggests:

Theorem 14.1 *The minimum Euclidean distance, r_{min}, over all particle pairs remains strictly greater than zero at the global minimum potential energy microcluster configuration.*

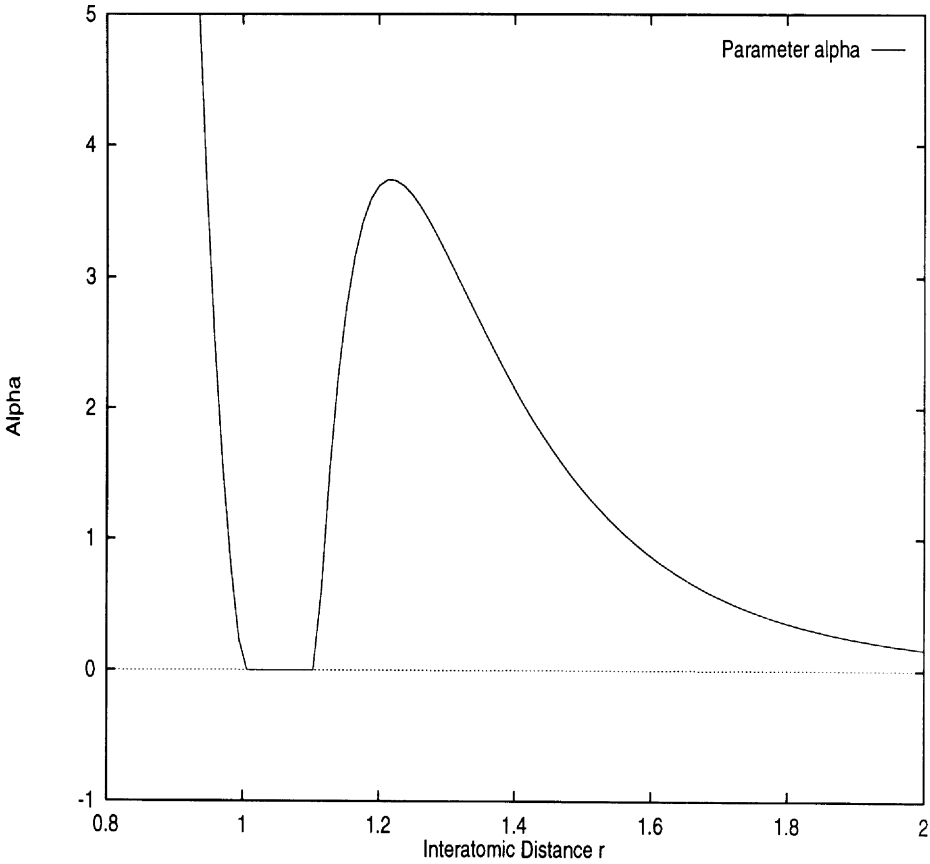


Figure 14.2. Evaluation of parameter α

Proof. Let $E^*(n)$ be the global minimum potential energy of an n–particle Lennard–Jones microcluster and $E^L(n), E^U(n)$ lower and upper bounds respectively of $E^*(n)$. By assuming that all particles are "touching" each other the following, second order in n, lower bound on $E^*(n)$ can be obtained:

$$E^L(n) = -\frac{n(n-1)}{2}.$$

A n-particle microcluster can be constructed by placing $n-3$ particles so that each of them touches three others, starting with three particles forming a equilateral triangle. This provides the following, linear in n, upper bound on $E^*(n)$:

$$E^U(n) = -3 - 3(n-3) = -3n + 6.$$

Let $r_{n-1,n}$ be the minimum interparticle distance of an n–particle microcluster at its global minimum potential energy configuration. Then, we have :

$$\begin{aligned}
-3n+6 &= E^U(n) \\
&\geq E^*(n) \\
&= \sum_{i=1}^{n-2}\sum_{j=i+1}^{n-1} v_{i,j} + \sum_{i=1}^{n-1} v_{i,n} \\
&\geq E^*(n-1) + \sum_{i=1}^{n-1} v_{i,n} \\
&= E^*(n-1) + \sum_{i=1}^{n-2} v_{i,n} + v_{n-1,n} \\
&\geq E^L(n-1) - (n-2) + v_{n-1,n}.
\end{aligned}$$

Solving for $v_{n-1,n}$ we get:

$$\begin{aligned}
v_{n-1,n} = \frac{1}{r_{min}^{12}} - \frac{2}{r_{min}^{6}} &\leq E^U(n) - E^L(n-1) + (n-2) \\
&= -3n+6+\frac{(n-1)(n-2)}{2} + (n-2) \\
&= \left(\frac{1}{2}n^2 - \frac{7}{2}n + 5\right).
\end{aligned}$$

Thus, for the minimum interparticle distance $r_{min}^{(I)}(n)$ of an n–particle microcluster we have:

$$r_{min}^{(I)}(n) = \left[\frac{-1+\sqrt{\frac{1}{2}n^2 - \frac{7}{2}n + 6}}{\frac{1}{2}n^2 - \frac{7}{2}n + 5}\right]^{\frac{1}{6}}.$$

Tighter lower bounds for $r_{n-1,n}$ can be obtained if instead of $E^U(n) = -3n+6$ we use the best reported solution in the literature as an upper bound on the global optimum solution $E^{opt}(n) \geq E^*(n)$.

$$\begin{aligned}
\frac{1}{r_{min}^{12}} - \frac{2}{r_{min}^{6}} &\leq E^{opt}(n) + \frac{(n-1)(n-2)}{2} + (n-2) \\
&= E^{opt}(n) + \frac{1}{2}n^2 - \frac{1}{2}n - 1.
\end{aligned}$$

After solving for $r_{min}^{(II)}(n)$ we get:

$$r_{min}^{(II)}(n) = \left[\frac{-1+\sqrt{E^{opt}(n) + \frac{1}{2}n^2 - \frac{1}{2}n}}{\frac{1}{2}n^2 - \frac{1}{2}n - 1}\right]^{\frac{1}{6}}.$$

n	$r_{min}^{(I)}$	$r_{min}^{(II)}$
3	1	1
4	1	1
5	0.8909	0.8949
6	0.8458	0.8505
7	0.8135	0.8272
8	0.7885	0.7984
9	0.7680	0.7798
10	0.7508	0.7626
11	0.7360	0.7479
12	0.7230	0.7359
13	0.7115	0.7265
14	0.7011	0.7137
15	0.6918	0.7033
16	0.6832	0.6940
17	0.6753	0.6854
18	0.6681	0.6780
19	0.6613	0.6714
20	0.6550	0.6640
21	0.6491	0.6579
22	0.6435	0.6518
23	0.6383	0.6468
24	0.6334	0.6414

Table 14.4. Lower bounds on r_{min}

Table 14.4 summarizes the results for $r_{min}^{(I)}(n)$ and $r_{min}^{(II)}(n)$ with $3 \leq n \leq 24$. These bounds appear to be much tighter than the ones reported by Xue *et al.* (1992).

Remark 14.3.2 A sufficient value of r_{min} for all practical purposes is about 0.95. If we define $\psi(r)$ to be equal to,

$$\psi(r) = \alpha\left(x_0^2 + y_0^2 + z_0^2 + x^2 + y^2 + z^2\right)$$

then the initial nonconvex expression for the pair potential has been transformed to the difference of two convex functions. More specifically,

$$v(r) = \phi(r) - \psi(r).$$

By applying the same analysis to every pair potential interaction, the total potential energy expression can be written as the difference of two convex functions (DC structure).

14.4 MATHEMATICAL MODEL

The problem of finding the global minimum energy of N particles interacting with central two–body forces can be formulated as the following nonlinear optimization problem :

$$
\begin{aligned}
\min_{x_i, y_i, z_i} \; V \quad &= \quad \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} v(x_i, y_i, z_i, x_j, y_j, z_j) \\
s.t. \quad & x_i^L \leq x_i \leq x_i^U, \qquad i = 2, \ldots, N \\
& y_i^L \leq y_i \leq y_i^U, \qquad i = 3, \ldots, N \\
& z_i^L \leq z_i \leq z_i^U, \qquad i = 4, \ldots, N \\
& x_1 = y_1 = z_1 = y_2 = z_2 = z_3 = 0
\end{aligned}
\tag{14.1}
$$

where $x_i^L, y_i^L, z_i^L, x_i^U, y_i^U, z_i^U$ define the box constraints for the x_i, y_i, z_i variables. In formulation (14.1), the objective function is nonconvex. Based on the transformation presented in the previous section (14.1) can be reformulated as the following (DC) programming problem involving an objective function which is the difference of two convex functions.

$$
\begin{aligned}
\min_{x_i,y_i,z_i} V \quad &= \quad \sum_{i=1}^{N-1}\sum_{j=i+1}^{N} v(x_i,y_i,z_i,x_j,y_j,z_j) \\
&+ \quad \sum_{i=1}^{N-1}\sum_{j=i+1}^{N} \alpha_{ij}(x_i^2+y_i^2+z_i^2+x_j^2+y_j^2+z_j^2) \\
&- \quad \sum_{i=1}^{N-1}\sum_{j=i+1}^{N} \alpha_{ij}(x_i^2+y_i^2+z_i^2+x_j^2+y_j^2+z_j^2) \\
s.t. \quad & x_i^L \le x_i \le x_i^U, \qquad i=2,\dots,N \\
& y_i^L \le y_i \le y_i^U, \qquad i=3,\dots,N \\
& z_i^L \le z_i \le z_i^U, \qquad i=4,\dots,N \\
& x_1 = y_1 = z_1 = y_2 = z_2 = z_3 = 0
\end{aligned}
\tag{14.2}
$$

$$
\text{where} \quad \alpha_{ij} = \max_{r_{ij}} \begin{cases} -\frac{v'(r_{ij})}{2r_{ij}} & \text{if } r_{min} \le r_{ij} \le r_0 \\ 0 & \text{if } r_0 \le r_{ij} \le r_1 \\ -\frac{v''(r_{ij})}{2} & \text{if } r_1 \le r_{ij} \le r_{max} \end{cases}
$$

$$
\text{and} \quad r_{ij} = \left[(x_i-x_j)^2+(y_i-y_j)^2+(z_i-z_j)^2\right]^{1/2}.
$$

In this section, an approach for finding the global minimum along the lines of the GOP algorithm, as described in Chapters 3-4 (see also Floudas and Visweswaran (1990, 1993) and Visweswaran and Floudas (1991; 1992a,b; 1993)), and the important concepts that form the basis of the αBB global optimization approach, as discussed in Chapters 10-11, are presented.

In view of the GOP requirements for convergence to the global minimum, the following transformation of variables is performed on formulation (14.2). The coordinate set of variables x_i, y_i, z_i *(x-type)* is "mirrored" to X_i, Y_i, Z_i *(y-type)* so that $x_i = X_i,\ y_i = Y_i,\ z_i = Z_i,\ \forall\, i = 1,\dots,N$. The expression for each pairwise interaction ij is augmented with the addition of the convexifying term in its quadratic form,

$$
\alpha_{ij}(X_i^2+Y_i^2+Z_i^2+X_j^2+Y_j^2+Z_j^2)
$$

and the subtraction of the convexifying term in its bilinear form,

$$
\alpha_{ij}(x_iX_i+y_iY_i+z_iZ_i+x_jX_j+y_jY_j+z_jZ_j)
$$

subject to the following equality constraints,

$$x_i = X_i,\ y_i = Y_i,\ z_i = Z_i,\ \ \forall\, i = 1, \ldots, N.$$

The effect of this transformation is to modify the nonconvexities appearing in the original objective function by transfering them into bilinear terms which are much easier to handle. Note that the terms that we subtract have the following bilinear form: $\alpha_{ij}(x_iX_i + y_iY_i + z_iZ_i + x_jX_j + y_jY_j + z_jZ_j)$. Observe that they are linear in x_i, y_i, z_i for fixed X_i, Y_i, Z_i and vice–versa. The values of α_{ij}'s are selected so that they are the minimum ones that guarantee convexification of each pair–potential term with the addition of the extra terms. Based on the above analysis, formulation (14.2) can be reformulated as follows:

$$\begin{aligned}
\min_{x_i,y_i,z_i} \quad V \quad &= \quad \sum_{i=1}^{N-1}\sum_{j=i+1}^{N} v(X_i, Y_i, Z_i, X_j, Y_j, Z_j) \\
&\quad + \sum_{i=1}^{N-1}\sum_{j=i+1}^{N} \alpha_{ij}(X_i^2 + Y_i^2 + Z_i^2 + X_j^2 + Y_j^2 + Z_j^2) \\
&\quad - \sum_{i=1}^{N-1}\sum_{j=i+1}^{N} \alpha_{ij}(x_iX_i + y_iY_i + z_iZ_i + x_jX_j + y_jY_j + z_jZ_j) \\
s.t. \quad & x_i = X_i, \qquad i = 2, \ldots, N \\
& y_i = Y_i, \qquad i = 3, \ldots, N \\
& z_i = Z_i, \qquad i = 4, \ldots, N \\
& x_i^L \le x_i \le x_i^U, \qquad i = 2, \ldots, N \\
& y_i^L \le y_i \le y_i^U, \qquad i = 3, \ldots, N \\
& z_i^L \le z_i \le z_i^U, \qquad i = 4, \ldots, N \\
& x_1 = y_1 = z_1 = y_2 = z_2 = z_3 = 0 \\
& x_i^L \le X_i \le x_i^U, \qquad i = 2, \ldots, N \\
& y_i^L \le Y_i \le y_i^U, \qquad i = 3, \ldots, N \\
& z_i^L \le Z_i \le z_i^U, \qquad i = 4, \ldots, N \\
& X_1 = Y_1 = Z_1 = Y_2 = Z_2 = Z_3 = 0.
\end{aligned} \tag{14.3}$$

Note that if we fix X_i, Y_i, Z_i and let x_i, y_i, z_i vary, formulation (14.3) corresponds to a linear programming problem. If we then fix x_i, y_i, z_i and let X_i, Y_i, Z_i vary (14.3) becomes a convex NLP programming problem. Thus, all requirements of the GOP algorithm are satisfied. Let us now consider a more restrictive version of (14.3), where all y–type variables are fixed, namely, $X_i = X_i^K, Y_i = Y_i^K, Z_i = Z_i^K$,

$$
\begin{aligned}
\min_{x_i,y_i,z_i} V \quad &= \sum_{i=1}^{N-1}\sum_{j=i+1}^{N} v(X_i^K, Y_i^K, Z_i^K, X_j^K, Y_j^K, Z_j^K) \\
&+ \sum_{i=1}^{N-1}\sum_{j=i+1}^{N} \alpha_{ij}\left[(X_i^K)^2 + (Y_i^K)^2 + (Z_i^K)^2\right] \\
&+ \sum_{i=1}^{N-1}\sum_{j=i+1}^{N} \alpha_{ij}\left[(X_j^K)^2 + (Y_j^K)^2 + (Z_j^K)^2\right] \\
&- \sum_{i=1}^{N-1}\sum_{j=i+1}^{N} \alpha_{ij}(x_iX_i^K + y_iY_i^K + z_iZ_i^K + x_jX_j^K + y_jY_j^K + z_jZ_j^K) \\
s.t. \quad & x_i = X_i^K, \qquad i = 2,\dots,N \qquad (14.4)\\
& y_i = Y_i^K, \qquad i = 3,\dots,N \\
& z_i = Z_i^K, \qquad i = 4,\dots,N \\
& x_i^L \le x_i \le x_i^U, \qquad i = 2,\dots,N \\
& y_i^L \le y_i \le y_i^U, \qquad i = 3,\dots,N \\
& z_i^L \le z_i \le z_i^U, \qquad i = 4,\dots,N \\
& x_1 = y_1 = z_1 = y_2 = z_2 = z_3 = 0.
\end{aligned}
$$

Problem (14.4) is called the *primal problem* and its solution provides an upper bound to the solution of (14.3) since it corresponds to a more restricted form. It is clear that (14.4) is always feasible and its solution corresponds to a single function evaluation. By applying the KKT conditions we obtain for the Langrange multipliers λ_{xi}, λ_{yi}, λ_{zi} associated with the equality constraints in (14.3) :

$$
\begin{aligned}
\lambda_{xi} &= (\sum_{j=1}^{N} \alpha_{ij})X_i^K, \qquad i = 2,\dots,N \\
\lambda_{yi} &= (\sum_{j=1}^{N} \alpha_{ij})Y_i^K, \qquad i = 3,\dots,N \\
\lambda_{zi} &= (\sum_{j=1}^{N} \alpha_{ij})Z_i^K, \qquad i = 4,\dots,N.
\end{aligned}
$$

By projecting at the space of the X_i, Y_i, Z_i variables and dualizing the equality constraints (Geoffrion (1972), Floudas (1995)), we can obtain the following relaxed dual formulation (see also the detailed analysis of Maranas and Floudas (1992)) :

$$
\begin{aligned}
& \min_{X_i, Y_i, Z_i, \mu_B} \mu_B \\
\text{s.t. to} \quad \mu_B \;\geq\; & L(x_i, y_i, z_i, X_i, Y_i, Z_i) = \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} v(X_i, Y_i, Z_i, X_j, Y_j, Z_j) \\
+ \; & \sum_{i=2}^{N} \left(\sum_{j=1}^{N} \alpha_{ij} \right) \min_{x_i} (X_i^K - X_i)(x_i - X_i) \\
+ \; & \sum_{i=3}^{N} \left(\sum_{j=1}^{N} \alpha_{ij} \right) \min_{y_i} (Y_i^K - Y_i)(y_i - Y_i) \\
+ \; & \sum_{i=4}^{N} \left(\sum_{j=1}^{N} \alpha_{ij} \right) \min_{z_i} (Z_i^K - Z_i)(z_i - Z_i) \\
& x_i^L \leq x_i \leq x_i^U, \qquad i = 2, \ldots, N \\
& y_i^L \leq y_i \leq y_i^U, \qquad i = 3, \ldots, N \\
& z_i^L \leq z_i \leq z_i^U, \qquad i = 4, \ldots, N \\
& x_1 = y_1 = z_1 = y_2 = z_2 = z_3 = 0 \\
& x_i^L \leq X_i \leq x_i^U, \qquad i = 2, \ldots, N \\
& y_i^L \leq Y_i \leq y_i^U, \qquad i = 3, \ldots, N \\
& z_i^L \leq Z_i \leq z_i^U, \qquad i = 4, \ldots, N \\
& X_1 = Y_1 = Z_1 = Y_2 = Z_2 = Z_3 = 0.
\end{aligned}
\tag{14.5}
$$

Problem (14.5) is linear in x_i, y_i, z_i so the minimum point x_i^*, y_i^*, z_i^* will be at an upper or lower bound depending on whether their coefficient is negative or positive, respectively. The variables x_i, y_i, z_i are called *connected variables*, and the gradients of the Lagrange function in terms of the connected variables are called *qualifying* constraints, which in problem (14.5), have the following form:

$$
\begin{aligned}
X_i^K &- X_i \\
Y_i^K &- Y_i \\
Z_i^K &- Z_i.
\end{aligned}
$$

This suggests that instead of minimizing explicitly in terms of the connected variables x_i, y_i, z_i, it is sufficient to solve (14.5) once for each combination of their bounds and then select the infimum over the calculated minima for all iterations. We have shown that the solution of (14.4) provides an upper bound and the solution of (14.5) a lower bound on the actual solution of the problem. This calls for an iterative scheme between (14.4) and (14.5) to determine the global solution of (14.1).

Based on the above analysis, the solution of the initial nonconvex problem (14.1) has been transformed to the solution of a series of convex nonlinear

relaxed dual problems which can be solved efficiently with existing algorithms (Murtagh and Saunders, 1983). The solution of the primal problem, which corresponds to a single function evaluation, provides an upper bound to the global minimum, whereas the relaxed dual problems yield lower bounds.

Remark 14.4.1 The main limitation of this approach is that for n connected variables, up to 2^n relaxed dual problems must be solved per iteration. The bounds of the variables are updated because this results in tighter lower bounding of the objective function. More specifically, the bounds of the variables for the current iteration are derived from the bounds of the variables of the relaxed dual problem, whose solution is the current point. Based on this, we can define a 2^n–tree with nodes corresponding to relaxed dual problems and levels corresponding to iterations. In the current iteration, only the Lagrange functions of relaxed dual problems which correspond to predecessor nodes for the node of the current relaxed dual problem are incorporated. It is interesting to note that their number is rather small, typically on the order of $\log_2(K)$. The bounds for the current iteration are provided by the relaxed dual problem corresponding to the parent node of the current relaxed dual problems. Note also that no qualifying constraints are needed in the formulation. The application of this property greatly reduces the required number of iterations for convergence.

14.5 ALGORITHMIC PROCEDURE

The basic steps of this procedure are as follows:

STEP 1 - *Initialization*

An initial point X_i^o, Y_i^o, Z_i^o is selected, the iteration counter K is set to zero, and a convergence tolerance ϵ is decided. Appropriate box constraints X_i^L,Y_i^L,Z_i^L,X_i^U,Y_i^U,Z_i^U for each variable x_i,y_i,z_i,X_i,Y_i,Z_i are selected. The minimum solution of the primal problems and the maximum solution of the relaxed dual problems so far are defined as P^{UBD} and R^{LBD}, respectively.

STEP 2 - *Primal problem*

The primal problem is solved at X_i^K, Y_i^K, Z_i^K for the variables x_i, y_i, z_i and the Lagrange multipliers $\lambda_{xi}, \lambda_{yi}, \lambda_{zi}$. This corresponds to a single function evaluation. If the solution of the primal is less than P^{UBD}, then P^{UBD} is updated to be the solution of the primal problem.

STEP 3 - *Selection of previous Lagrange functions and update of bounds*

The Lagrange functions from relaxed dual problems in previous iterations ($k = 1, 2, \ldots, K-1$) corresponding to predecessor nodes for the current iteration are included to be constraints in the current iteration's relaxed dual problems. The box constraints of the current iteration are the ones of the relaxed dual problem corresponding to the parent node for the current tree level (iteration).

STEP 4 - *Update of α_{ij} parameters*

The convexification parameters α_{ij} are updated in every iteration as follows:

$$\alpha_{ij} = \max_{r_{ij}} \begin{cases} -\frac{v'(r_{ij})}{2r_{ij}} & \text{if } r_{min} \leq r_{ij} \leq r_0 \\ 0 & \text{if } r_0 \leq r_{ij} \leq r_1 \\ -\frac{v''(r_{ij})}{2} & \text{if } r_1 \leq r_{ij} \leq r_{max} \end{cases}$$

$$\begin{aligned} & r_{ij} = \left[(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2\right]^{\frac{1}{2}} \\ & x_i^l \leq x_i \leq x_i^u, \qquad i = 2, \ldots, N \\ & y_i^l \leq y_i \leq y_i^u, \qquad i = 3, \ldots, N \\ & z_i^l \leq z_i \leq z_i^u, \qquad i = 4, \ldots, N \\ & x_1 = y_1 = z_1 = y_2 = z_2 = z_3 = 0. \end{aligned}$$

Here $x_i^l, y_i^l, z_i^l, x_i^u, y_i^u, z_i^u$ are the current bounds on x_i, y_i, z_i.

STEP 5 - *Non-interior current point* (X_i^K, Y_i^K, Z_i^K)

If any of the X_i^K, Y_i^K, Z_i^K are at a bound (lower/upper), then the appropriate relaxed dual problems are eliminated.

STEP 6 - *Solution of relaxed dual problems*

For every combination of bounds $B_{xi}^K, B_{yi}^K, B_{zi}^K$ of the connected variables x_i, y_i, z_i respectively, that has not been found to activate the checks in **STEP 5** the following relaxed dual problem is solved :

$$\min_{X_i, Y_i, Z_i, \mu_B} \mu_B$$

subject to

$$\begin{aligned} \mu_B &\geq L_k(x_i^{B_{xi}^k}, y_i^{B_{yi}^k}, z_i^{B_{zi}^k}, X_i, Y_i, Z_i), \quad \forall \quad k = 1, 2, \ldots, K-1 \\ \mu_B &\geq L_K(x_i^{B_{xi}^K}, y_i^{B_{yi}^K}, z_i^{B_{zi}^K}, X_i, Y_i, Z_i) \\ & x_i^{K,l} \leq X_i \leq x_i^{K,u}, \qquad i = 2, \ldots, N \\ & y_i^{K,l} \leq Y_i \leq y_i^{K,u}, \qquad i = 3, \ldots, N \\ & z_i^{K,l} \leq Z_i \leq z_i^{K,u}, \qquad i = 4, \ldots, N \\ & X_1 = Y_1 = Z_1 = Y_2 = Z_2 = Z_3 = 0. \end{aligned}$$

Here L_k are the Lagrange functions from the previous iterations, and L_K the current one. $B_{xi}^k, B_{yi}^k, B_{zi}^k$ are the set of values of the connected variables x_i, y_i, z_i in the Lagrange function from the kth iteration and $x_i^{K,l}, y_i^{K,l}, z_i^{K,l}, x_i^{K,u}$, $y_i^{K,u}, z_i^{K,u}$ are the bounds on X_i, Y_i, Z_i in the current iteration K.

STEP 8 - *Update* $R^{LBD}, X_i^K, Y_i^K, Z_i^K$

After all relaxed dual problems have been solved, a new lower bound R^{LBD} for the global minimum is selected which corresponds to the lowest value of the stored solutions of all relaxed dual problems from all iterations so far. At the same time, the values of X_i, Y_i, Z_i of the minimum stored solution are selected for the next current point. Once selected, the stored solution is deleted from

the list. This ensures that no relaxed dual problem will return to the same solution during successive iterations.

STEP 9 - *Check for convergence*

If $P^{UBD} - R^{LBD} < 0$, then rigorous convergence has been achieved and the algorithm stops. If $P^{UBD} - R^{LBD} < \epsilon$, then ϵ–convergence has been achieved and the algorithm is stopped. Otherwise, set $K = K + 1$ and return to **STEP 2**.

This global optimization algorithmic procedure has been applied to a number of problems which are presented in the next section.

14.6 SMALL CLUSTER EXAMPLES

By using the global optimization procedure presented above, the global minimum potential energy configurations of small microclusters ($2 \leq N \leq 13$) are generated. Two different potential expressions have been employed, the scaled Lennard–Jones potential and the Morse potential, because of their simplicity and wide acceptance. The value of the parameter a for the Morse potential was set to be $a = 3.0$. Table 14.5 summarizes the total minimum potential energies for Lennard–Jones and Morse microclusters. The potential energy of Morse microclusters for the same number of particles is always less than the energy of Lennard–Jones microclusters. This is consistent with the plot of Figure 14.1 which illustrates that the expression for the Morse potential ($a = 3.00$) is always less than the one for the scaled Lennard–Jones potential. The global minimum structure for $N = 2$ corresponds to two particles "touching" each other. For $N = 3$ three particles form a unit equilateral triangle as the global minimum structure. For $N = 4$ the four particles are placed at the vertices of a regular tetrahedron. For $N = 5$ a trigonal bipyramid, slightly contracted along the symmetry axis and distended in the symmetry plane corresponds to the global minimum energy structure. For $N = 6$ a regular octahedron with slightly contracted sides yields the global optimum configuration. For $N = 7$ the pentagonal bipyramid with slightly distended edges and contracted axial distances represents the lowest energy structure. From $N = 8$ to $N = 13$ the structure involving the global minimum total potential energy is obtained by adding particles at the face centered pockets of the seven particle microcluster until the icosahedral structure is obtained for $N = 13$.

For a comprehensive report on the cluster minima the reader is directed to the Cambdridge Cluster Database (http://brian.ch.cam.ac.uk/CCD.html) for Lennard–Jones clusters, C_{60} clusters, TIP4P water clusters, Morse clusters, and Sutton-Chen clusters among others, as well as the recent book on test problems for local and global optimization by Floudas *et al.* (1999) that describes mathematical models and results on clusters that include Lennard–Jones, Morse, Tersoff, Brenner, and Bolding-Andersen potentials.

For larger N, the complexity of the problem limits the use of the general global optimization procedure. In the next section, a "relaxation" of the global optimization procedure is presented which, for larger microclusters, yields tight

N	V_{total}^{LJ}	V_{total}^{Morse}
2	-1.000	-1.000
3	-3.000	-3.000
4	-6.000	-6.000
5	-9.104	-9.300
6	-12.712	-13.544
7	-16.505	-17.552
8	-19.822	-22.043
9	-24.113	-26.778
10	-28.423	-31.889
11	-32.766	-37.931
12	-37.968	-44.098
13	-44.327	-51.737

Table 14.5. Global minimum potential energies $2 \leq N \leq 13$ for Lennard–Jones and Morse microclusters

lower and upper bounds on the global minimum total potential energy as well as excellent initial points for a possible local optimization approach.

14.7 A RELAXATION OF THE GLOBAL OPTIMIZATION APPROACH

As has been mentioned earlier, the bottleneck of the presented global optimization approach is the large number of connected variables which gives rise to a potentially very high number of relaxed dual problems that have to be solved in each iteration. This motivates the following relaxation of the global optimization approach which stems from the fact that the global minimum potential energy configuration of an N–particle cluster is usually composed by a slight perturbation in the global minimum configuration for $N-1$ particles plus an extra particle. This means that when solving for the global minimum structure of N particles, and the global minimum configuration for $N-1$ particles is known, tight bounds for the coordinate variables of the first $N-1$ particles can be used around the global minimum positions of an $(N-1)$ par-

ticle microcluster and only for the Nth particle "loose" bounds are required. It must be noted that this relaxation procedure cannot guarantee convergence to the global minimum; however, computational experience for small microclusters demonstrates the efficiency of the approach to locate the best known microcluster conformation.

Based on this, the global minimum potential energy problem in (14.1) can be reformulated as:

$$
\begin{aligned}
\min_{x_i,y_i,z_i} \quad V \quad &= \quad \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} v(x_i, y_i, z_i, x_j, y_j, z_j) \\
\text{st} \quad & x_i^* - e \leq x_i \leq x_i^* + e, \qquad i = 2, \ldots, N-1 \\
& y_i^* - e \leq y_i \leq y_i^* + e, \qquad i = 3, \ldots, N-1 \\
& z_i^* - e \leq z_i \leq z_i^* + e, \qquad i = 4, \ldots, N-1 \\
& -E \leq x_N \leq E \\
& -E \leq y_N \leq E \\
& -E \leq z_N \leq E \\
& x_1 = y_1 = z_1 = y_2 = z_2 = z_3 = 0.
\end{aligned}
$$

Here $e = 0.01 - 0.05$ accounts for the perturbation around the nominal point x_i^*, y_i^*, z_i^* which corresponds to the global minimum potential energy configuration of an $N-1$ microcluster. Also, $E = 2.0 - 4.0$ defines a cube where the Nth particle is expected to vary. Since x_i^*, y_i^*, z_i^* corresponds to the global minimum potential energy point of an $N-1$ microcluster, V in this formulation must be convex in terms of x_i, y_i, z_i for $i = 1, \ldots, N-1$ in a neighborhood of x_i^*, y_i^*, z_i^*. By assuming that e in this formulation is sufficiently small, then in order to transform V to the difference of two convex functions only the term $\sum_{j=1}^{N} \alpha_{i,j} \left(x_N^2 + y_N^2 + z_N^2\right)$ is required. Thus, the Lagrange function takes the following form:

$$
\begin{aligned}
L(x_i, y_i, z_i, X_i, Y_i, Z_i) &= \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} v(X_i, Y_i, Z_i, X_j, Y_j, Z_j) \\
&+ \left(\sum_{i=1}^{N} \alpha_{i,N} \right) (X_N^K - X_N)(x_N - X_N) \\
&+ \left(\sum_{i=1}^{N} \alpha_{i,N} \right) (Y_N^K - Y_N)(y_N - Y_N) \\
&+ \left(\sum_{i=1}^{N} \alpha_{i,N} \right) (Z_N^K - Z_N)(z_N - Z_N).
\end{aligned}
$$

Note that this formulation involves only three connected variables x_N, y_N, z_N which means that irrespective of how large N is, only up to $2^3 = 8$ relaxed dual problems must be solved per iteration. By combining the Lagrange function with the presented global optimization procedure a number of examples $8 \leq N \leq 24$, have been considered for the scaled Lennard–Jones potential. Tight lower and upper bounds on the global minimum solutions were first established by selecting the tolerance to be between $0.01 - 0.05$. Then, by switching to a local optimization algorithm, the global minimum potential energy structures were found. These results are summarized in Table 14.6, where E^L, E^U correspond to the lower and upper bounds on the global minimum energy E^* and Iter is the total number of iterations needed for obtaining the bounds.

A further relaxation of this procedure can be realized by "fixing" the first $N-1$ particles to the coordinates of the global minimum total potential energy configuration of an $N-1$ microcluster. This implies that the Lagrange function now becomes a function of only three variables X_N, Y_N, Z_N.

$$
\begin{aligned}
L(X_N, Y_N, Z_N) &= \sum_{i=1}^{N-1} v(X_i^*, Y_i^*, Z_i^*, X_N, Y_N, Z_N) \\
&+ \left(\sum_{i=1}^{N} \alpha_{i,N} \right) (X_N^K - X_N)(x_N - X_N) \\
&+ \left(\sum_{i=1}^{N} \alpha_{i,N} \right) (Y_N^K - Y_N)(y_N - Y_N) \\
&+ \left(\sum_{i=1}^{N} \alpha_{i,N} \right) (Z_N^K - Z_N)(z_N - Z_N) \\
&+ \sum_{i=1}^{N-2} \sum_{j=i+1}^{N-1} v(X_i^*, Y_i^*, Z_i^*, X_j^*, Y_j^*, Z_j^*).
\end{aligned}
$$

N	E^L	E^U	e	E	Iter	E^*
8	-20.633	-19.683	0.01	1.0	12	-19.822
9	-28.321	-24.111	0.01	1.0	21	-24.113
10	-29.783	-28.326	0.01	1.5	26	-28.423
11	-34.233	-32.559	0.02	1.5	31	-32.766
12	-39.522	-37.557	0.02	1.5	29	-37.968
13	-44.487	-42.290	0.02	1.5	15	-44.327
14	-49.469	-47.218	0.02	1.5	10	-47.845
15	-52.655	-52.083	0.02	1.5	11	-52.323
16	-57.612	-56.601	0.02	1.5	21	-56.816
17	-62.463	-61.222	0.05	1.5	32	-61.318
18	-67.591	-65.615	0.05	1.5	20	-66.531
19	-73.567	-72.117	0.10	1.5	22	-72.660
20	-78.485	-76.972	0.10	1.5	21	-77.177
21	-83.265	-81.427	0.10	1.5	14	-81.685
22	-87.754	-86.116	0.10	1.5	16	-86.810
23	-92.949	-91.276	0.10	1.5	8	-92.844
24	-98.920	-96.997	0.10	1.5	19	-97.349

Table 14.6. Bounds and global minimum potential energies for LJ microclusters

Note that the point X_i^*, Y_i^*, Z_i^* corresponds to the global minimum for an $N-1$ microcluster. This new formulation results in substantial reduction in the CPU time per relaxed dual problem due to the reduction in the number of variables. Therefore one can quickly generate points that are quite close to the global minimum one. This calls for a two–stage procedure where the relaxed global optimization procedure generates a configuration which is likely to be close to the global minimum one and then a local optimization algorithm "relaxes" this configuration to the nearest minimum one. By applying this procedure for Morse microclusters $8 \leq N \leq 30$ a number of minima have been obtained and are summarized in Table 14.7, where E^{init} is the initial point provided by the relaxed global optimization procedure, E^{found} the obtained solution from the local optimization algorithm, and E^{best} the best reported value in the literature. It appears that this procedure is successful at generating all best known configurations except for $N = 18$ because in this case more than one particle has to be relocated. In fact, by fixing only $N - 2$ particles this time and applying the relaxed global optimization procedure on the last two, the global minimum structures for $N = 18$ has also been reproduced.

N	E^{init}	E^{found}	E^{best}
8	-21.494	-22.043	-22.043
9	-26.307	-26.778	-26.778
10	-31.149	-31.274	-31.889
11	-36.350	-37.931	-37.931
12	-42.605	-44.098	-44.098
13	-51.653	-51.737	-51.737
14	-56.366	-56.660	-56.755
15	-62.746	-63.162	-63.162
16	-68.863	-69.036	-69.036
17	-75.041	-75.662	-75.662
18	-81.855	-82.579	-82.579
19	-89.087	-90.647	-90.647
20	-96.605	-97.417	-97.417
21	-103.960	-104.337	-104.337
22	-111.741	-112.041	-112.041
23	-120.546	-120.787	-120.787
24	-127.642	-127.885	-127.885
25	-135.272	-136.073	-136.073
26	-144.696	-145.322	-145.322
27	-152.161	-152.514	-152.514
28	-160.294	-160.773	-160.773
29	-169.909	-170.115	-170.115
30	-177.285	-177.579	-177.579

Table 14.7. Minimum potential energies of Morse ($a = 3$) microclusters

Although these simplifications cannot guarantee convergence to the global minimum, it appears that the second relaxation approach behaves very well for microclusters $N \leq 30$. Note that these approaches can be readily extended to larger microclusters by allowing more than one particle to have "loose" coordinate bounds. In this case, if M particles involve "loose" coordinate bounds then up to 2^{3M} relaxed dual problems must be solved per iteration.

14.8 DISCUSSION ON MAGIC NUMBERS AND INTERATOMIC SPACING

Figure 14.3 shows the difference in the total potential energy between the global minimum potential energy configurations of two microclusters which differ only by a single particle. Both the scaled Lennard–Jones and the Morse potential cases are plotted. Note that the differences in energy between the two consecutive configurations are larger for the Morse microclusters than for the Lennard–Jones microclusters. This is expected because the expression for the Morse potential is always larger in absolute value then that of the scaled Lennard–Jones potential. Both curves clearly exhibit an increasing trend which is consistent with the fact that the extra potential interactions following the addition of a particle become larger as the size of the original microcluster increases. However, this trend is not monotonically increasing. Both curves exhibit a saw–tooth behavior which reflects the fact that certain microcluster sizes are more "stable" than others, indicating the presence of the so–called "magic numbers". For both the scaled Lennard–Jones and the Morse potential the incremental potential energy decrease curve (see Figure 14.3) exhibits peaks for $N = 13, 19$ and 23. This can be explained by examining the 3D plots of the scaled Lennard–Jones microclusters for $N = 13, 19$ and 23. For $N = 13$ the global minimum potential energy structure is a compact icosahedral structure. For $N = 19$ the optimal structure involves a compact icosahedral core composed of 13 particles, a five–fold ring located at the FC empty lattice sites of the icosahedral core, and a final particle in contact with one particle from the icosahedral core and all five–fold ring particles. For $N = 23$ we have a compact icosahedral core, two interpenetrating five fold rings located at the FC empty lattice sites of the icosahedral core and two extra particles in contact with all five–fold ring particles. The common element of these three structures is that *they are derived from the previous structure by the addition of an extra particle on top of a five fold ring.* Because the additional particle is "in contact" with six particles, the associated potential energy reduction is very high.

As shown earlier, the global minimum potential energy configuration of an $N = 13$ particle Lennard–Jones microcluster is an icosahedron about a central atom. Due to the symmetry of the structure, all interatomic spacings between the twelve atoms occupying the icosahedral lattice sites are equal to 1.0134 (slight dilation). Furthermore, the interatomic distances between the central atom and the twelve atoms which occupy the icosahedral sites are all equal to 0.9638 (slight compression). This implies that there exist only two discrete values for the interatomic spacings in a $N = 13$ Lennard–Jones microcluster. Figure 14.4 shows the interatomic spacings, sorted in increasing order, of only the 13 atoms which form the first icosahedral layer in a $N = 13, 14$ and 24 Lennard–Jones microcluster. Note that for $N = 13$ there exist only two discrete values for the interatomic spacing. However, for $N = 14$ the symmetry breaks down and the interatomic spacings assume more than two values. For $N = 24$ the interatomic spacing "spreads" over a much larger range of values. By averaging these interatomic spacings, we find that the mean interatomic spacing

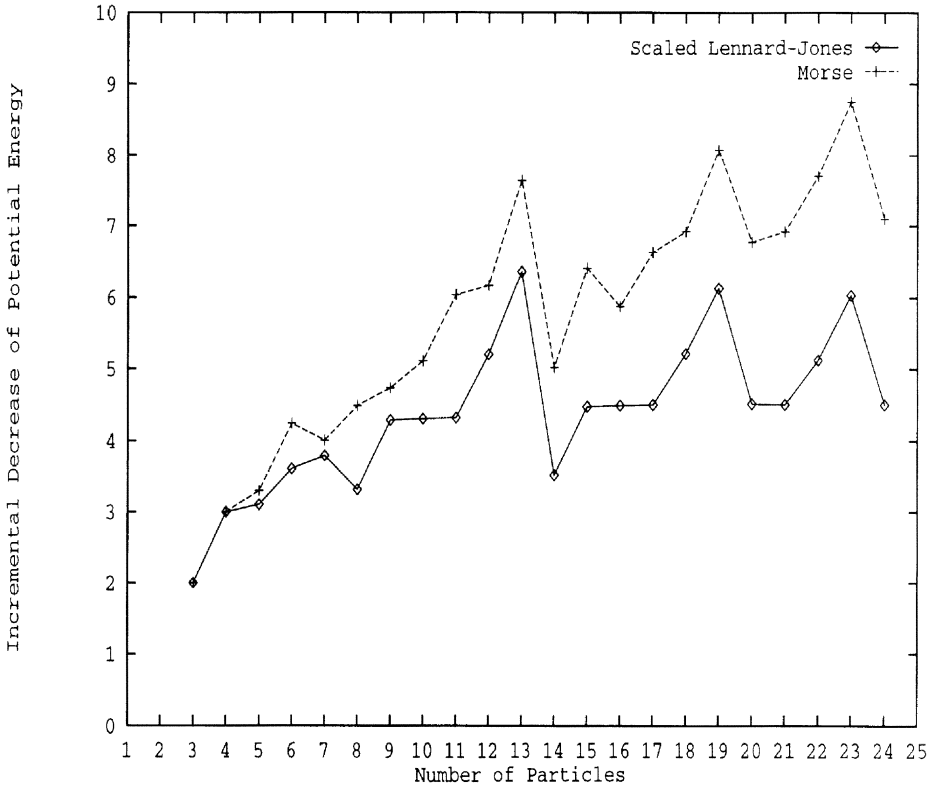


Figure 14.3. Incremental decrease of the total potential energy.

for $N = 13$ is equal to 0.9992, for $N = 14$ this spacing reduces to 0.9988 and for $N = 24$ it further decreases to 0.9957. Therefore, as the microcluster size increases the original icosahedral core progressively not only distorts, but also shrinks in size.

14.9 SUMMARY

In this chapter an overview of the minimum total potential energy of microclusters problem has been presented and a global optimization approach was introduced for finding the global minimum potential energy configuration of small microclusters interacting with central forces (Maranas and Floudas (1992,1993)). It involved the transformation of the initial nonconvex total potential energy expression to the difference of two convex functions (DC) through a novel eigenvalue analysis and the application of a Primal–Relaxed Dual global optimization approach to the resulting formulation. Furthermore, the approach

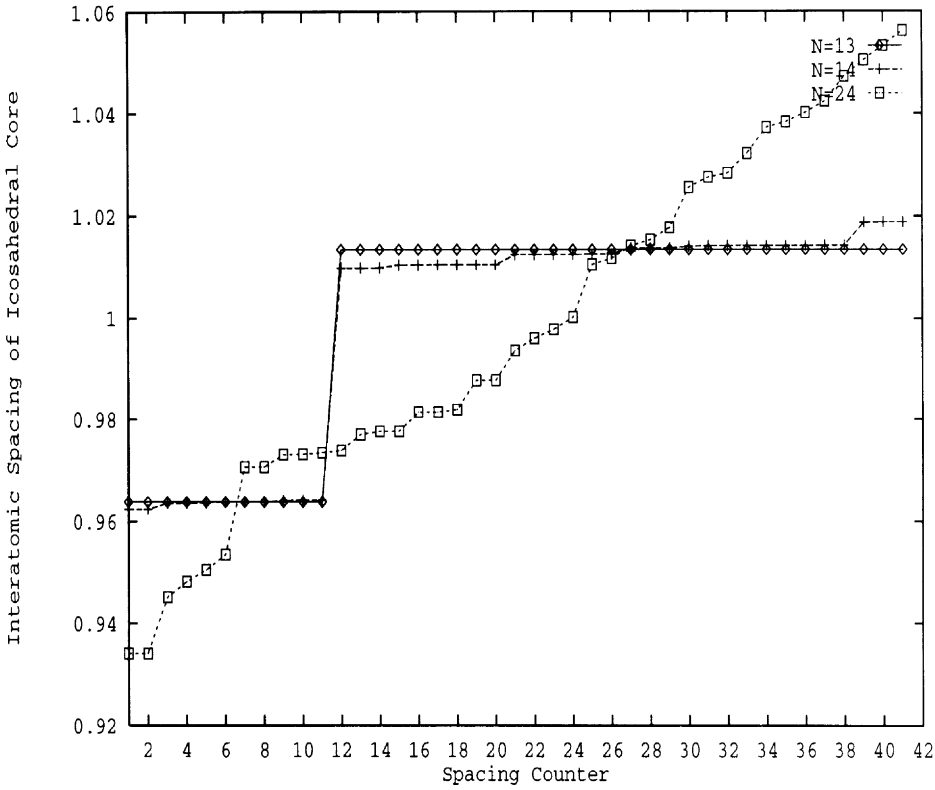


Figure 14.4. Interatomic spacings in the original icosahedral core of N=13,14 and 24 particle LJ microclusters.

is guaranteed to converge to the global optimum in a finite number of iterations. For larger microclusters, two different relaxations of this approach yielded tight lower and upper bounds on the global minimum,as well as initial points very close to the global optimum. Global optimization test problems for several other classes of clusters can be found in the recent book by Floudas *et al.* (1999).

15 THE αBB APPROACH IN MOLECULAR STRUCTURE PREDICTION

In this chapter, we discuss the application of the αBB approach to the structure prediction of small molecules in which the bond lengths and the bond angles are assumed to be fixed at their equilibrium values. Section 15.1 presents an introduction and background to this class of problems. Section 15.2 defines the problem in Cartesian coordinates. Section 15.3 presents the reformulation of the problem in the internal coordinate system. Section 15.4 discusses the lower bounding function which forms the basis for the application of the αBB to box constrained nonlinear optimization problems, as discussed in Chapter 11. Finally, section 15.5 discusses the application of the αBB to the structure prediction of propanal. The material presented in this chapter is based on the work of Maranas and Floudas (1994a), (1994b).

15.1 INTRODUCTION AND BACKGROUND

The search for the global minimum potential energy conformation of a molecule faces the plethora of minima in the multivariable potential energy hyperspace, making it a very challenging and complex problem. Although the basic building blocks of molecules, that is, atoms, remain virtually unchanged in different compounds, the versatility of the ways in which they can be combined and reconfigured results in numerous conformations for a given molecule. One of these conformations, the most stable one, is of particular importance because it dictates most of the properties of the molecule. Experimental evidence (Troyer

C. A. Floudas, *Deterministic Global Optimization*
© Springer Science+Business Media Dordrecht 2000

and Cohen, 1991; Levitt, 1982) shows that in the great majority of cases the global minimum potential energy conformation is an excellent candidate for being the most stable conformation.

The most accurate representation of the atomic interactions in a molecule is the *ab initio* quantum mechanical approach. Quantum mechanical calculations, however, are not tractable for all but the simplest molecules. As a result, simplified potential energy expressions that adequately capture the energetics of various types of atomic interactions must be derived. In doing so, trade–offs between maximizing the accuracy of the potential energy expression and, at the same time, minimizing the computational effort to evaluate these functions have to be addressed.

A series of such simplifications starts with the Born–Oppenheimer approximation, which dictates that the Schrödinger equation for the molecule can be separated into two parts. One part describes the motions of the electrons and the other determines the motion of the nuclei. Therefore, these two sets of motions can be studied independently. This implies that the potential energy of a molecule in the *ground electronic state* is a function of only the nuclear positions (Burkert and Allinger, 1982). This defines a potential energy multi–dimensional hypersurface in the coordinate space of the nuclear positions, which is simply called the potential energy surface, and encompasses the impact on the potential energy of all possible conformations that a molecule can assume. Each point on the potential energy surface corresponds to a different *conformation* of the molecule and local minimum points on this surface are referred to as *conformers*. The identification of the global minimum point on this surface is a very important objective of molecular mechanics (Burkert and Allinger, 1982).

Molecular mechanics calculations employ an empirically derived set of potential energy contributions for approximating these atomic interactions. This set of potential energy contributions, called the *force field*, contains adjustable parameters that are selected in a such a way as to provide the best possible agreement with experimental data. The main assumption introduced in molecular mechanics is that every parameter is associated with *a specific interaction* rather than a specific molecule. These potential model parameters can be bond lengths, covalent bond angles, bond stretching, bending, or rotating constants, non–bonded atom interaction constants, etc. Molecular mechanics force fields include two–body, three–body or even n–body interaction terms.

$$V = \sum_{i>j} V_{ij} + \sum_{i>j>k} V_{ijk} + \sum_{i>j>k>l} V_{ijkl} + \quad \ldots \tag{15.1}$$

It can be seen that the total potential energy V is not simply the sum of all pair interactions; it may involve energy terms which depend on the position and properties of three or even more atoms due to the fact that the charges on the particles are in general polarized. However, for fairly nonpolar atoms these terms can be neglected with little effect on the total potential energy (Hopfinger, 1973). This means that for nonpolar molecules the total potential

energy expression is given approximately by,

$$V = \sum_{i>j} V_{ij}(B) + \sum_{i>j} V_{ij}(NB) \tag{15.2}$$

where B indicates bonded atom interactions and NB nonbonded atom interactions. In most conformational calculations the bonded atom interactions are assumed to be constant and independent of the actual conformation and thus they can be omitted. Also, in general only *pairwise additive terms* are significant in conformational calculations, although higher order terms can become important under certain conditions. Finally, by assuming that covalent bond lengths and angles remain constant at their equilibrium values, $1-2$ and $1-3$ atomic interactions can also be omitted.

In practice, the pairwise potential interactions are usually represented by Lennard–Jones or Buckingham type classical empirical potential functions:

Lennard–Jones 6–12 Potential Function

$$v_{ij} = \frac{B_{ij}}{{r_{ij}}^{12}} - \frac{A_{ij}}{{r_{ij}}^{6}}.$$

The attractive term $-A_{ij}/r^6$ is rigorously derived for a pair of identical spherically symmetrical and chemically saturated molecules, and $B_{ij}/{r_{ij}}^{12}$ is an approximation for the repulsive component for which no rigorously derived expression exists.

Buckingham Potential Function

$$v_{ij} = A_{ij} \exp(B_{ij} r_{ij}) + \frac{C_{ij}}{r_{ij}^6}.$$

The Buckingham potential function differs from the Lennard–Jones function only in the form of the repulsive term. A two–parameter exponential function is used which in principle should be more specific in describing the repulsive interaction because of the additional parameter.

A very large number of methods have been proposed for finding the most stable conformation of a molecule through the identification of the global minimum point on the potential energy surface. The great majority of existing methods attempt to locate this point by tracing either deterministically or stochastically single or multiple paths on the potential energy surface, under the conjecture that some of them will converge to the global minimum point.

Despite the plethora of the currently available methods and the intriguing ways that they propose to circumvent the existence of the myriad of local minima on the potential energy hypersurface, little progress has been made towards proving convergence to the global minimum. The key limitation is that unless there is a single potential well, the obtained minimum energy conformation depends heavily on the supplied initial conformation. This is why in practice many trial geometries serve as initial points for the employed optimization method.

15.2 PROBLEM DEFINITION

The simplifications employed herein start with the well established conjecture in molecular mechanics that the most energetically stable conformation of the molecule is the one that corresponds to the global minimum potential energy. Tractable expressions for this potential energy function are estimated based on the Born–Oppenheimer approximation. Because the most stable conformations of fairly nonpolar molecules are studied in this work, only pairwise, additive, two–body interaction terms are considered.

The expression for the total potential energy V of a molecule involves only the sum of a number of pairwise potential interaction terms. Each pairwise potential term is a function of only the Euclidean distances, r_{ij}, between the interacting atoms i and j which in turn are directly related to the Cartesian coordinates of the atoms forming the molecule. Therefore, V can be fully represented in the coordinate space x_i, y_i, z_i of the atomic coordinates.

Note, however, that a number of equality constraints must be included in the formulation to reflect the fact that all covalent bond lengths and covalent bond angles are assumed to be fixed at their equilibrium values. From a mathematical standpoint this is the key difference between the molecular conformation problem and the microcluster configuration problem. After defining $\mathcal{B}$ to be the set of bonded atoms and $\mathcal{NB}$ the set of non–bonded atoms, V can be expressed in the Cartesian atomic coordinate space as follows:

$$
\begin{aligned}
V &= \sum_{\substack{i<j \\ ij \in \mathcal{NB}}} v(r_{ij}) \\
\text{s.t.} & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad (15.3) \\
r_{ij} &= \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2}, \quad \forall\, ij \in \mathcal{NB} \\
r^o_{ij} &= \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2}, \quad \forall\, ij \in \mathcal{B} \\
\theta^o_{ijk} &= -\frac{(x_i - x_j)(x_j - x_k) + (y_i - y_j)(y_j - y_k) + (z_i - z_j)(z_j - z_k)}{r^o_{ij} r^o_{jk}}, \\
& \quad \forall\ ijk \text{ such that } ij,\, jk \in \mathcal{B} \\
\text{and} & \qquad x_1 = y_1 = z_1 = y_2 = z_2 = z_3 = 0
\end{aligned}
$$

where V is the total potential energy of the molecule; $v(r_{ij})$ is the pairwise potential expression; r_{ij} is the Euclidean distance between nonbonded atom i with atom j; r^o_{ij}, θ^o_{ijk} are the fixed covalent bond lengths and angles respectively; and x_i, y_i, z_i are the atomic Cartesian coordinates. Note that in the summation $i < j$ is used so that we avoid doublecounting pair interactions and the interaction of an atom with itself. Furthermore, by imposing

$x_1 = y_1 = z_1 = y_2 = z_2 = z_3 = 0$ we eliminate the translational and rotational degrees of freedom of the molecule.

Remark 15.2.1 The use of Cartesian coordinates in local optimization techniques greatly facilitates the calculation of the total potential energy V. However, by employing Cartesian coordinates we introduce a number of complex, highly nonlinear equality constraints, (one for each r_{ij}^o and θ_{ijk}^o), which are very difficult to handle in the context of a deterministic global optimization algorithm. On the other hand, internal coordinates depend on the covalent bond lengths r_{ij}, covalent bond angles θ_{ijk} and dihedral angles ϕ_{ijkl}.

Remark 15.2.2 Since the covalent bond lengths and bond angles are assumed to be fixed at their equilibrium values, the internal coordinates reduce to only dihedral (torsion) angles. Furthermore, at least for acyclic molecules, by selecting one torsion angle per rotational degree of freedom, a set of independent torsion angles, t_k, is defined which enables the description of the rest of the torsion angles as linear combinations. In polycyclic molecules, however, it becomes more complicated to define a set of independent internal coordinates because the bond and torsion angles are correlated. In principle, the independent set of torsion angles is much smaller than the set of atomic coordinates. This is a considerable advantage for global optimization algorithms because the number of variables is typically the bottleneck of the computational effort.

Based on these observations, independent internal coordinates rather than Cartesian coordinates are employed. The total potential energy can then be written using the set of independent internal coordinates as follows:

$$V = \sum_{\substack{i<j \\ ij \in \mathcal{NB}}} v\left(r_{ij}(t_k)\right) \tag{15.4}$$

where $t_k \in$ set of independent internal coordinates, $k = 1, \ldots, K$.

Remark 15.2.3 The above formulation does not involve any equality constraints; however, the functionality between r_{ij} and the set of independent internal variables (independent torsion angles) needs to be established.

In the next section, an efficient procedure for obtaining explicit expressions for the interatomic Euclidean distances r_{ij} as functions of internal coordinates is discussed. This enables the problem reformulation in the internal coordinate space.

15.3 PROBLEM REFORMULATION IN INTERNAL COORDINATES

Although it is quite straightforward to express all r_{ij}'s in the Cartesian coordinate set, unfortunately this is not the case when r_{ij} must be expressed as a function of an independent set of internal coordinates. In doing so it is first necessary to establish a connecting path, formed by a sequence of covalent bonds, between every atom i and atom j. For all pairs of atoms (i, j) in a molecule

there is always an acyclic chain of atoms connected by covalent bonds, which links atom i with atom j. The connectivity of atoms in a molecule always defines a *undirected connected graph* (Cormen *et al.*, 1989). This introduces a natural way of *partitioning* the set of nonbonded r_{ij}'s according to the minimum number of atoms participating in an acyclic chain connecting atom i with atom j. The aforementioned partitioning of r_{ij}'s can be realized with the definition of the following sets:

$$\begin{aligned}
\mathcal{B} &= \{\,(i,j)\ :\ \text{i,j bonded atoms}\} \\
\mathcal{NB} &= \{\,(i,j)\ :\ \text{i,j nonbonded atoms}\} \\
\mathcal{B}^2 &= \{\,(i,j)\ :\ i<j,\ \text{and}\ (i,j)\ \in\ \mathcal{B}\} \\
\vdots &= \qquad\qquad \vdots \\
\mathcal{B}^m &= \{\,(i_1,i_2,\ldots,i_m)\ :\ i_1<i_m, \text{and}\ (i_1,i_2)\in\mathcal{B},\ \ldots,\ (i_{m-1},i_m)\in\mathcal{B}, \\
&\qquad \text{where}\ (i_1,i_2,\ldots,i_m)\ \text{isanacyclicchain}\}\,.
\end{aligned}$$

Note that $\mathcal{B}$ is the set of pairs of atoms which are connected with covalent bonds and $\mathcal{NB}$ is the set of pairs of atoms which are not. Also, $\mathcal{B}^2,\ldots,\mathcal{B}^m$, correspond to *directed* sets of chains of covalently bonded atoms involving two, three, and m atoms respectively.

If only a single atom (2) is between atoms (1) and (3) such that $(1,2,3)\ \in\ \mathcal{B}^3$, then it is well known that,

$$r_{13}^2 \quad = \quad r_{12}^2 + r_{23}^2 - 2r_{12}r_{23}\cos(\theta_{123})$$

where θ_{123} is the angle formed by covalent bonds (1-2) and (2-3). However, when more than one atom is required to connect atoms i and j the derivation of r_{ij} is much more involved (see Figure 15.1). Let us now consider three atoms, numbered (1), (2), and (3) such that atom (1) is bonded with atom (2), and atom (2) is bonded with atom (3). The bond lengths r_{12} and r_{23} as well as the covalent bond angle θ_{123} are given. First, we select a base coordinate system defined by the positions of the first three atoms, (1), (2), and (3) such that the origin is chosen at atom (1), the negative x–axis passes through atom (2), and atom (3) lies in the first or second quadrant of the xy plane (See Figure 15.1). It is easy to show that the Cartesian coordinates of these three atoms are:

$$\begin{pmatrix}
x_1 = 0, & y_1 = 0, & z_1 = 0 \\
x_2 = -r_{12}, & y_2 = 0, & z_2 = 0 \\
x_3 = r_{23}\cos(\theta_{123}) - r_{12}, & y_3 = r_{23}\sin(\theta_{123}), & z_3 = 0
\end{pmatrix}$$

The position of an additional atom (4), bonded to atom (3), can be fully specified if the length of the bond r_{34}, the bond angle θ_{234}, and the dihedral (torsion) angle ϕ_{1234} are known. ϕ_{1234} is the angle between the plane defined by atoms (1),(2),(3) and the plane defined by atoms (2),(3),(4). The coordinates of atom (4) after rotational transformation and a translation are given by Thompson (1969),

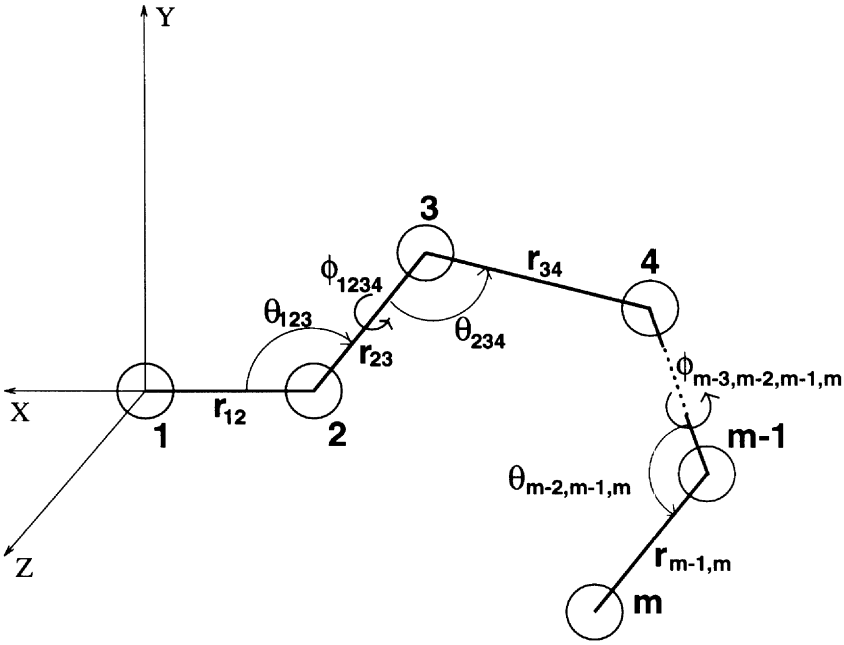


Figure 15.1. Coordinate set of atomic chain.

$$\begin{bmatrix} x_4 \\ y_4 \\ z_4 \\ 1 \end{bmatrix} = \mathbf{B_2}\,\mathbf{B_3}\,\mathbf{B_4} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

where $\mathbf{B_2}, \mathbf{B_3}$ and $\mathbf{B_4}$ are the following 4×4 transformation matrices whose elements are products of covalent bond lengths with trigonometric functions of covalent and dihedral angles :

$$\mathbf{B_2} = \begin{bmatrix} -1 & 0 & 0 & -r_{12} \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

$$\mathbf{B_3} = \begin{bmatrix} -\cos(\theta_{123}) & -\sin(\theta_{123}) & 0 & -r_{23}\cos(\theta_{123}) \\ \sin(\theta_{123}) & -\cos(\theta_{123}) & 0 & r_{23}\sin(\theta_{123}) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

$$\mathbf{B_4} = \begin{bmatrix} -\cos(\theta_{234}) & -\sin(\theta_{234}) & 0 & b_1 \\ \sin(\theta_{234})\cos(\phi_{1234}) & -\cos(\theta_{234})\cos(\phi_{1234}) & -\sin(\phi_{1234}) & b_2 \\ \sin(\theta_{234})\sin(\phi_{1234}) & -\cos(\theta_{234})\sin(\phi_{1234}) & \cos(\phi_{1234}) & b_3 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

where b_1, b_2, b_3 are :

$$\begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} = \begin{bmatrix} -r_{34}\cos(\theta_{234}) \\ r_{34}\sin(\theta_{234})\cos(\phi_{1234}) \\ r_{34}\sin(\theta_{234})\sin(\phi_{1234}) \end{bmatrix},$$

By repetitively placing extra atoms in this four–atom chain, an m–atom chain composed of atoms connected sequentially with covalent bonds is formed. By generalizing the presented analysis the coordinates of the mth atom in the coordinate system that the first three atoms define are (Thompson, 1969):

$$\begin{bmatrix} x_m \\ y_m \\ z_m \\ 1 \end{bmatrix} = \mathbf{B_2}\,\mathbf{B_3}\cdots\mathbf{B_m}\begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix},$$

where $\mathbf{B_m}$ is defined as,

$$\mathbf{B_m} = \begin{bmatrix} -\cos(\theta_m) & -\sin(\theta_m) & 0 & -r_m\cos(\theta_m) \\ \sin(\theta_m)\cos(\phi_m) & -\cos(\theta_m)\cos(\phi_m) & -\sin(\phi_m) & r_m\sin(\theta_m)\cos(\phi_m) \\ \sin(\theta_m)\sin(\phi_m) & -\cos(\theta_m)\sin(\phi_m) & \cos(\phi_m) & r_m\sin(\theta_m)\sin(\phi_m) \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

following the simplified notation,

$$r_{m-1,m} \longleftarrow r_m, \quad \theta_{m-2,m-1,m} \longleftarrow \theta_m, \quad \phi_{m-3,m-2,m-1,m} \longleftarrow \phi_m.$$

Based on the above analysis, explicit expressions for the Cartesian coordinates x_m, y_m, z_m of the mth atom in a m–atom chain can be obtained as functions of the bond lengths, covalent bond angles, and torsion angles. Since $x_1 = y_1 = z_1 = 0$, the squared Euclidean distance r_{1m}^2 is,

$$r_{1m}^2 \;=\; x_m^2 + y_m^2 + z_m^2.$$

After expanding the squared terms, the expression for r_{1m}^2 becomes the sum of a very large number of terms involving the product of bond lengths with sines and/or cosines of covalent bond lengths and/or dihedral (torsion) angles. Simplified explicit expressions for r_{1m}^2, $m = 2, \ldots, 6$ as functions of ϕ_{ijkl}

$$r_{1m}^2 \;=\; r_{1m}^2(\phi_{ijkl},\; ijkl \;\in\; \mathcal{B}^4)$$

are obtained via symbolic computations and are given in section 15.4.1.

Clearly, not all dihedral angles are independent. In fact, for acyclic molecules, a set of independent dihedral angles can be defined involving one independent dihedral angle per rotational degree of freedom. For example, propane–like molecules require only three independent dihedral angles in order to fully specify all other dihedral angles and therefore the molecular conformation. In general, for each dihedral angle ϕ_{ijkl} of an acyclic molecule, an independent dihedral angle t_k can be defined so that,

$$\phi_{ijkl} \;=\; t_k + \omega_{ijkl}$$

where ω_{ijkl} counts how many degrees ϕ_{ijkl} is lagging or is ahead of t_k. This means that all nonbonded atom distances r_{ij}, and hence the expression for the total potential energy V, can be expressed as a function of only these independent dihedral angles, one for each rotational degree of freedom.

$$V \;=\; \sum_{j=4}^{m} \;\; \sum_{(i_1,\ldots,i_j)\,\in\,\mathcal{B}^j} \; \upsilon\left(r_{i_1,i_j}\right)$$

$$\text{where} \qquad r_{i_1,i_j} \;=\; r_{i_1,i_j}\left(\phi_{ijkl},\; (i,j,k,l) \;\in\; \mathcal{B}^4\right),$$
$$\forall\,(i_1,\ldots,i_j) \;\in\; \mathcal{B}^j,\;\; j = 4,\ldots,m$$

$$\text{and} \qquad \forall\,(i,j,k,l) \;\in\; \mathcal{B}^4,\;\; \exists\, k^*,\; \omega_{ijkl} \;\text{ such that }\; \phi_{ijkl} \;=\; t_{k^*} + \omega_{ijkl}$$

where m is the number of atoms in the longest chain connecting two interacting atoms; r_{i_1,i_j} is the expression derived earlier for the Euclidean distance between the two end atoms i_1 and i_j; and $\upsilon(r_{i_1,i_j})$ is the pairwise potential interaction function. Note that j starts from $(j = 4)$ because there must be at least two atoms between the interacting atoms before considering their contribution to the total potential energy of the molecule. This is because for $j = 2, 3$ the Euclidean distances between the end atoms i_1 and i_j are functions of only the fixed covalent bond angles and covalent bond lengths.

This implies that the problem of minimizing the total potential energy of an acyclic molecule can be formulated as an essentially unconstrained optimization problem in the space of independent dihedral angles. The expression, however, for the total potential energy function is nonconvex involving a plethora of local minima; therefore existing optimization algorithms cannot consistently locate the global minimum. To address this problem, the deterministic branch and bound global optimization algorithm, αBB, is introduced to locate the global minimum solution by constructing converging lower and upper bounds. These bounds are successively refined by iteratively partitioning the initial feasible region into subregions. Upper bounds to the global minimum can be obtained by simply calculating the value of V at some point. Lower bounds, however, are more difficult to derive. In the next section, a convex lower bounding function is proposed and its properties are analyzed.

15.4 CONVEX LOWER BOUNDING FUNCTION L OF V

In the previous section, explicit relations for the nonbonded atom distances have been derived as functions of the dihedral (torsion) angles ϕ_{ijkl}. However, not all of these dihedral angles are independent. In fact, there are only as many independent dihedral angles as the number of rotational degrees of freedom. For example, for ethane–like molecules all dihedral angles can be expressed as functions of a *single* dihedral angle. For propane–like molecules two dihedral angles are required for fully specifying the conformation of the molecule. This means that all nonbonded distances r_{ij} and consequently the expression for the total potential energy V can be expressed as a function of only these independent dihedral angles, defined one for each rotational degree of freedom :

$$V = V(t_k), \quad k = 1, \ldots, K \tag{15.5}$$

where t_k's correspond to the K independent dihedral angles.

The next step in this analysis is to postulate a convex lower bounding function $\mathcal{L}$ of V as discussed in Chapter 11.

$$\mathcal{L} = V + \alpha \sum_{k=1}^{K} (t_k^L - t_k)(t_k^U - t_k) \tag{15.6}$$

$$\text{where} \qquad \alpha \geq \max\left(0, -\frac{1}{2} \min_{\substack{k \\ t_k^L \leq t_k \leq t_k^U}} \lambda_k^V\right).$$

Note that t_k^L, t_k^U correspond to the lower and upper bounds on t_k which are typically set to $t_k^L = 0$ and $t_k^U = 2\pi$. Also, α is a nonnegative parameter which must be greater or equal to the negative one half of the minimum eigenvalue of V over $t_k^L \leq t_k \leq t_k^U$.

Based on the analysis and the important properties of the αBB global optimization approach presented in Chapter 11, the problem of minimizing the

total potential energy of a molecule can now be explicitly formulated in the coordinate space of independent dihedral (torsion) angles t_k.

$$\min_{t_k} \quad V = \sum_{j=4}^{m} \sum_{(i_1,\ldots,i_j) \in \mathcal{B}^j} v\left(r_{i_1,i_j}\right)$$

$$\text{where} \quad r_{i_1,i_j} = r_{i_1,i_j}\left(\phi_{ijkl},\ (i,j,k,l) \in \mathcal{B}^4\right), \quad \forall\,(i_1,\ldots,i_j) \in \mathcal{B}^j,\ j = 4,\ldots,m$$

$$\text{and} \quad \forall\,(i,j,k,l) \in \mathcal{B}^4,\ \exists\, k,\ \omega_{ijkl} \text{ such that } \phi_{ijkl} = t_k + \omega_{ijkl}$$

$$t_k^L \le t_k \le t_k^U, \quad k = 1,\ldots,K.$$

Note that V is the total potential energy composed by the sum of all pairwise potential interactions $v(r_{i_1,i_j})$ which in turn are functions of only the corresponding interatomic distances r_{i_1,i_j}. These distances are defined over all unique acyclic j–atom chains $(i_1,\ldots,i_j)$ belonging to $\mathcal{B}^j$. Note also that j spans from $j = 4$, avoiding constant distance atom chains for $j = 2,3$, to $j = m$ where m is the longest acyclic chain in the molecule. Explicit expressions for the interatomic Euclidean distances r_{i_1,i_j} for all acyclic atom chains $[(i_1,\ldots,i_j),\ j = 4,\ldots,m]$ in terms of dihedral angles ϕ_{ijkl}, $(i,j,k,l) \in \mathcal{B}^4$ are presented in the next subsection.

15.4.1 *Analytical Expressions for $r_{ij} = r_{ij}(\phi_{ijkl})$*

Based on the analysis of molecular conformations, the following general expressions for the Euclidean distances between the end atoms in m–atom ($m \le 6$) chains are obtained:

$$\begin{aligned}
r_{ik}^2 &= r_{ij}^2 + r_{jk}^2 - 2\cos(\theta_{ijk}) r_{ij} r_{jk}, \quad (i,j,k) \in \mathcal{B}^3 \\
r_{il}^2 &= r_{ij}^2 + r_{jk}^2 + r_{kl}^2 \\
&- 2\cos(\theta_{ijk}) r_{ij} r_{jk} - 2\cos(\theta_{jkl}) r_{jk} r_{kl} \\
&- 2\left[\sin(\theta_{ijk})\sin(\theta_{jkl})\cos(\phi_{ijkl}) - \cos(\theta_{ijk})\cos(\theta_{jkl})\right] r_{ij} r_{kl}, \\
&\quad (i,j,k,l) \in \mathcal{B}^4
\end{aligned}$$

$$
\begin{aligned}
r_{im}^2 &= r_{ij}^2 + r_{jk}^2 + r_{kl}^2 + r_{lm}^2 \\
&- 2\cos(\theta_{ijk}) r_{ij} r_{jk} - 2\cos(\theta_{jkl}) r_{jk} r_{kl} - 2\cos(\theta_{klm}) r_{kl} r_{lm} \\
&- 2\left[\sin(\theta_{ijk})\sin(\theta_{jkl})\cos(\phi_{ijkl}) - \cos(\theta_{ijk})\cos(\theta_{jkl})\right] r_{ij} r_{kl} \\
&- 2\left[\sin(\theta_{jkl})\sin(\theta_{klm})\cos(\phi_{jklm}) - \cos(\theta_{jkl})\cos(\theta_{klm})\right] r_{jk} r_{lm} \\
&- 2\left[\cos(\theta_{ijk})\cos(\theta_{jkl})\cos(\theta_{klm})\right. \\
&\quad - \sin(\theta_{ijk})\sin(\theta_{jkl})\cos(\theta_{klm})\cos(\phi_{ijkl}) \\
&\quad - \sin(\theta_{ijk})\cos(\theta_{jkl})\sin(\theta_{klm})\cos(\phi_{ijkl})\cos(\phi_{jklm}) \\
&\quad - \cos(\theta_{ijk})\sin(\theta_{jkl})\sin(\theta_{klm})\cos(\phi_{jklm}) \\
&\quad \left. - \sin(\theta_{ijk})\sin(\theta_{klm})\sin(\phi_{ijkl})\sin(\phi_{jklm})\right] r_{ij} r_{lm}, \\
&\quad (i,j,k,l,m) \in \mathcal{B}^5 \\
r_{in}^2 &= r_{ij}^2 + r_{jk}^2 + r_{kl}^2 + r_{lm}^2 + r_{mn}^2 \\
&- 2\cos(\theta_{ijk}) r_{ij} r_{jk} - 2\cos(\theta_{jkl}) r_{jk} r_{kl} \\
&- 2\cos(\theta_{klm}) r_{kl} r_{lm} - 2\cos(\theta_{lmn}) r_{lm} r_{mn} \\
&- 2\left[\sin(\theta_{ijk})\sin(\theta_{jkl})\cos(\phi_{ijkl}) - \cos(\theta_{ijk})\cos(\theta_{jkl})\right] r_{ij} r_{kl} \\
&- 2\left[\sin(\theta_{jkl})\sin(\theta_{klm})\cos(\phi_{jklm}) - \cos(\theta_{jkl})\cos(\theta_{klm})\right] r_{jk} r_{lm} \\
&- 2\left[\sin(\theta_{klm})\sin(\theta_{lmn})\cos(\phi_{klmn}) - \cos(\theta_{klm})\cos(\theta_{lmn})\right] r_{kl} r_{mn} \\
&- 2\left[\cos(\theta_{ijk})\cos(\theta_{jkl})\cos(\theta_{klm})\right. \\
&\quad - \sin(\theta_{ijk})\sin(\theta_{jkl})\cos(\theta_{klm})\cos(\phi_{ijkl}) \\
&\quad - \sin(\theta_{ijk})\cos(\theta_{jkl})\sin(\theta_{klm})\cos(\phi_{ijkl})\cos(\phi_{jklm}) \\
&\quad - \cos(\theta_{ijk})\sin(\theta_{jkl})\sin(\theta_{klm})\cos(\phi_{jklm}) \\
&\quad \left. - \sin(\theta_{ijk})\sin(\theta_{klm})\sin(\phi_{ijkl})\sin(\phi_{jklm})\right] r_{ij} r_{lm} \\
&- 2\left[\cos(\theta_{jkl})\cos(\theta_{klm})\cos(\theta_{lmn})\right. \\
&\quad - \sin(\theta_{jkl})\sin(\theta_{klm})\cos(\theta_{lmn})\cos(\phi_{jklm}) \\
&\quad - \sin(\theta_{jkl})\cos(\theta_{klm})\sin(\theta_{lmn})\cos(\phi_{jklm})\cos(\phi_{klmn}) \\
&\quad - \cos(\theta_{jkl})\sin(\theta_{klm})\sin(\theta_{lmn})\cos(\phi_{klmn}) \\
&\quad \left. - \sin(\theta_{jkl})\sin(\theta_{lmn})\sin(\phi_{jklm})\sin(\phi_{klmn})\right] r_{jk} r_{mn} \\
&+ 2\left[\cos(\theta_{ijk})\cos(\theta_{jkl})\cos(\theta_{klm})\cos(\theta_{lmn})\right. \\
&\quad - \sin(\theta_{ijk})\sin(\theta_{jkl})\cos(\theta_{klm})\cos(\theta_{lmn})\cos(\phi_{ijkl}) \\
&\quad - \sin(\theta_{ijk})\cos(\theta_{jkl})\sin(\theta_{klm})\cos(\theta_{lmn})\cos(\phi_{ijkl})\cos(\phi_{jklm}) \\
&\quad - \cos(\theta_{ijk})\sin(\theta_{jkl})\sin(\theta_{klm})\cos(\theta_{lmn})\cos(\phi_{jklm}) \\
&\quad - \sin(\theta_{ijk})\cos(\theta_{jkl})\cos(\theta_{klm})\sin(\theta_{lmn})\cos(\phi_{ijkl})\cos(\phi_{jklm})\cos(\phi_{klmn}) \\
&\quad - \cos(\theta_{ijk})\sin(\theta_{jkl})\cos(\theta_{klm})\sin(\theta_{lmn})\cos(\phi_{jklm})\cos(\phi_{klmn}) \\
&\quad - \cos(\theta_{ijk})\cos(\theta_{jkl})\sin(\theta_{klm})\sin(\theta_{lmn})\cos(\phi_{klmn}) \\
&\quad + \sin(\theta_{ijk})\sin(\theta_{jkl})\sin(\theta_{klm})\sin(\theta_{lmn})\cos(\phi_{ijkl})\cos(\phi_{klmn}) \\
&\quad - \sin(\theta_{ijk})\sin(\theta_{klm})\cos(\theta_{lmn})\sin(\phi_{ijkl})\sin(\phi_{jklm}) \\
&\quad - \sin(\theta_{ijk})\cos(\theta_{klm})\sin(\theta_{lmn})\sin(\phi_{ijkl})\sin(\phi_{jklm})\cos(\phi_{klmn}) \\
&\quad + \sin(\theta_{ijk})\sin(\theta_{lmn})\sin(\phi_{ijkl})\cos(\phi_{jklm})\sin(\phi_{klmn}) \\
&\quad - \sin(\theta_{ijk})\cos(\theta_{jkl})\sin(\theta_{lmn})\cos(\phi_{ijkl})\sin(\phi_{jklm})\sin(\phi_{klmn}) \\
&\quad \left. - \cos(\theta_{ijk})\sin(\theta_{jkl})\sin(\theta_{lmn})\sin(\phi_{jklm})\sin(\phi_{klmn})\right] r_{ij} r_{mn} \\
&\quad (i,j,k,l,m,n) \in \mathcal{B}^6
\end{aligned}
$$

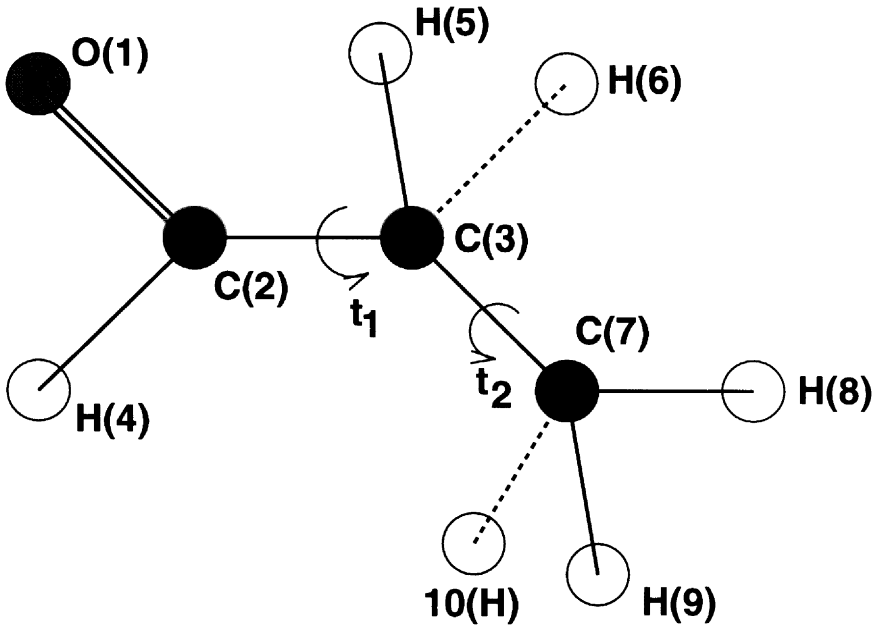


Figure 15.2. Propanal molecule.

Based on the topology of the molecule, an independent set of dihedral angles t_k, $k = 1, \ldots, K$ can be selected so that each dihedral angle ϕ_{ijkl} differs by a constant factor ω_{ijkl} from some independent dihedral angle t_k. Finally, lower and upper bounds t_k^L, t_k^U (typically 0 and 2π) are provided for the independent dihedral angles, t_k.

15.5 COMPUTATIONAL STUDY : PROPANAL

This example is the problem of finding the global minimum potential energy conformation of the propanal molecule (CH_3CH_2CHO) (See Figure 15.2). This molecule involves fifteen chains of four covalently bonded atoms and six chains of five covalently bonded atoms. The values of the covalent bond lengths and

atom–atom	r_o ($\mathring{A}$)
C=O	1.208
C–C	1.509
C-H(ald)	1.113
C-H	1.113

Table 15.1. Covalent bond lengths r_o in the propanal example

atom–atom–atom	θ_o
O=C–H(ald)	120.00°
O=C–C	120.00°
H(ald)–C–C	120.00°
C–C–C	109.50°
C–C–H	109.50°
H–C–H	109.50°

Table 15.2. Covalent bond angles θ_o in the propanal example

covalent bond angles are assumed to be fixed at their equilibrium values and they are given in Table 15.1 and Table 15.2 respectively as proposed by Susnow *et al.* (1991).

The Buckingham potential function was selected to model the non–bonded atom interactions.

$$v(r_{ij}) \quad = \quad A_{ij} \exp\left(B_{ij} r_{ij}\right) \; + \; \frac{C_{ij}}{r_{ij}^6}$$

The values of the interaction parameters A_{ij}, B_{ij}, and C_{ij} are given in Table 15.3 (Susnow *et al.*, 1991). This problem involves two independent internal coordinates. One possible selection can be the dihedral angle t_1 between the planes defined by O(1)=C(2)-C(3) and C(2)–C(3)–H(5), and t_2 between the

atom–atom	A $(kcal/mol)$	B $(\mathring{A}^{-1})$	C $(kcal\ \mathring{A}^6/mol)$
H–O	16152.0	-3.85800	-144.970
H(ald)–C	15628.0	-3.43410	-282.030
H–C	13188.0	-3.63370	-169.550
H(ald)–H	13630.0	-4.16677	-77.0920
H–H	13630.0	-4.16677	-77.0920

Table 15.3. Data for parameters A_{ij}, B_{ij}, and C_{ij} in the propanal example

planes H(5)–C(3)–C(7) and C(3)–C(7)–H(8). The bounds on t_1, t_2 are :

$$\begin{aligned} 0 &\leq t_1 \leq 2\pi \\ 0 &\leq t_2 \leq \frac{2\pi}{3}. \end{aligned}$$

All other dihedral angles are :

$$\begin{aligned}
\phi_{O(1)=C(2)-C(3)-H(5)} &= t_1 \\
\phi_{O(1)=C(2)-C(3)-H(6)} &= t_1 + \tfrac{2\pi}{3} \\
\phi_{O(1)=C(2)-C(3)-C(7)} &= t_1 - \tfrac{2\pi}{3} \\
\phi_{C(4)-C(2)-C(3)-H(5)} &= t_1 + \pi \\
\phi_{C(4)-C(2)-C(3)-H(6)} &= t_1 - \tfrac{\pi}{3} \\
\phi_{C(4)-C(2)-C(3)-C(7)} &= t_1 + \tfrac{\pi}{3}
\end{aligned}$$

$$\begin{aligned}
\phi_{H(5)-C(3)-C(7)-H(8)} &= t_2 \\
\phi_{H(5)-C(3)-C(7)-H(9)} &= t_2 + \tfrac{2\pi}{3} \\
\phi_{H(5)-C(3)-C(7)-H(10)} &= t_2 - \tfrac{2\pi}{3} \\
\phi_{H(6)-C(3)-C(7)-H(8)} &= t_2 - \tfrac{2\pi}{3} \\
\phi_{H(6)-C(3)-C(7)-H(9)} &= t_2 \\
\phi_{H(6)-C(3)-C(7)-H(10)} &= t_2 + \tfrac{2\pi}{3} \\
\phi_{C(2)-C(3)-C(7)-H(8)} &= t_2 + \tfrac{2\pi}{3} \\
\phi_{C(2)-C(3)-C(7)-H(9)} &= t_2 - \tfrac{2\pi}{3} \\
\phi_{C(2)-C(3)-C(7)-H(10)} &= t_2.
\end{aligned}$$

For the given set of data the following solutions have been found by repeatedly solving the problem of minimizing the total potential energy with the local optimization solver MINOS5.4 (Murtagh and Saunders, 1983) using randomly selected starting points.

$$\begin{aligned}
t_1 &= 41.17^\circ, & t_2 &= 59.84^\circ & \longrightarrow\ V_{min} &= 0.769101 \text{ kcal/mol} \\
t_1 &= 198.83^\circ, & t_2 &= 60.16^\circ & \longrightarrow\ V_{min} &= 0.769101 \text{ kcal/mol} \\
t_1 &= 300.00^\circ, & t_2 &= 60.00^\circ & \longrightarrow\ V_{min} &= 0.830866 \text{ kcal/mol} \\
t_1 &= 360.00^\circ, & t_2 &= 60.06^\circ & \longrightarrow\ V_{min} &= 0.844303 \text{ kcal/mol.}
\end{aligned}$$

t_2 appears to be approximately equal to 60 degrees in all local configurations implying that the hydrogens connected with carbon atom $C(3)$ repel the one connected with carbon atom $C(7)$. Furthermore, t_1 assumes a value of 41.17

degrees in one of the two global minimum potential energy conformations in an attempt to most effectively balance the attractions of the oxygen atom $O(1)$ with the rest of the hydrogen atoms. The proposed global optimization algorithm with $\alpha = 5$ and tolerance $\epsilon = 10^{-4}$ converged in all 100 runs to either of the two multiple global minimum solutions,

$$(41.17°, 59.84°) \text{ or } (198.83°, 60.16°) \longrightarrow V^* = 0.769101 \text{ kcal/mol}$$

in about 400 iterations with an average CPU time of 80 seconds on a HP-9000/730 workstation. For $\alpha = 1$ convergence to the global minimum is attained in all 100 hundred runs in about 110 iterations and required only 20 seconds of CPU time.

Summary

In this chapter, the deterministic branch and bound global optimization algorithm αBB was applied for finding the most stable conformations of molecules based solely on the energetics of non–bonded interactions (Maranas and Floudas (1994a,b)). First, explicit relations for the non–bonded Euclidean distances as function of bond lengths, covalent bond angles, and dihedral (torsion) angles were derived. Then, the problem was formulated as an unconstrained nonconvex optimization problem on a set of independent dihedral angles. Based on an eigenvalue analysis, a convex lower bounding function $\mathcal{L}$ of the total potential energy function V was defined. The properties of function $\mathcal{L}$, as discussed in Chapter 11, coupled with an efficient partitioning scheme enabled us to construct a global optimization algorithm guaranteed to always possess ϵ–converge to the global minimum conformation through the solution of a series of convex nonlinear optimization problems. Finally, the proposed approach was applied to the case study of propanal.

16 THE αBB APPROACH IN PROTEIN FOLDING

In this chapter, we address the structure prediction problem in protein folding via the αBB global optimization approach. Section 16.1 provides an introduction to the protein folding problem. Section 16.2 describes the potential energy model and two solvation models based on the area and volume accessible to the solvent. Section 16.3 discusses the concepts, the algorithmic issues and the implementation of the global optimization approach αBB as it is applied to the protein folding problem. Finally, section 16.4 presents computational studies for unsolvated and solvated pentapeptides. The material presented in this chapter is based on the work of Maranas *et al.* (1996), Androulakis *et al.* (1997a), Klepeis *et al.* (1998), Klepeis and Floudas (1999a), and Floudas *et al.* (1999b).

16.1 INTRODUCTION

Proteins are the most complex and vital molecules in nature. This complexity arises from an intricate balance of intra- and inter-molecular interactions which define the native three-dimensional structure of the system, and subsequently its biological functionality. Anfinsen's hypothesis (Anfinsen *et al.*, 1961) suggests that this native structure is in a state of thermodynamic equilibrium with the lowest free energy conformation. Experimental studies have since shown that, under native physiological conditions and after denaturation, globular proteins spontaneously refold to their unique, native structure (Kim and Bald-

C. A. Floudas, *Deterministic Global Optimization*
© Springer Science+Business Media Dordrecht 2000

win, 1990). Understanding the transition of a protein from a disordered state to its native state defines the protein folding problem.

The use of computational techniques and simulations in addressing the protein folding problem became possible through the introduction of qualitative and quantitative methods for modeling these systems. The development of realistic energy models established a link to the field of global optimization. Since the number of local minima is vast, the corresponding problem formulation is classified as the *multiple-minima* problem. The basis for these difficulties is best summarized by Levinthal's paradox (Levinthal (1968), Dill and Chan (1997)). This paradox suggests a contradiction between the almost infinite number of possible stable states that the system may sample and the relatively short time scale required for actual protein folding. In contradiction to Anfinsen's thermodynamic hypothesis, Levinthal's observations suggest that the native state is the lowest kinetically accessible free energy minimum, which may be different from the true global minimum. These principles have been used to develop computational techniques for predicting protein folding pathways (Church *et al.*, 1996; Leopold *et al.*, 1992; Šali *et al.*, 1996). Such techniques attempt to map the shape of the energy hypersurface and determine whether this surface directs a protein towards a dominant conformational basin.

Based on the complexity of the energy hypersurface, there is an obvious need for the development of efficient global optimization techniques. Although the energy can be expressed analytically, exhaustive searches are possible for only the smallest of systems. These observations, and the importance of the protein folding problem, have propelled the introduction of new global search strategies specifically designed for these problems. A number of proceedings and reviews have focused on global optimization issues in molecular systems and include Leach (1991), Scheraga (1992), Pardalos *et al.* (1994), Vásquez *et al.* (1994), Pardalos *et al.* (1996), Neumaier (1997b), Chan and Dill (1998), and Floudas *et al.* (1999b).

16.2 MATHEMATICAL MODELING

16.2.1 Protein Representation

From a chemical point of view, a protein is a polymer chain composed of a sequence of various amino acid residues connected by peptide bonds. Naturally occurring proteins are composed of 20 different amino acid residues, where the form of the side chain (e.g., methyl, butyl, benzoic, etc.) defines these residues. This basic structure is slightly different only in the case of proline residues.

The repeating unit -$NC_{\alpha}C$'- connected by peptide bonds defines the backbone of the protein. In addition, the protein possesses amino and carboxyl end groups. Covalent bond angle requirements and interatomic forces bend and twist the chain in a characteristic way for each protein. The protein chain configures into a unique three-dimensional geometric conformation called the folded state of the protein. It is this configuration which defines the shape of

the protein surface, as well as the specific chemically active groups present on the surface, which in turn sets the biological functionality of the protein.

Instead of specifying the coordinate vector for all atoms in a protein molecule, one can specify all bond lengths, covalent bond angles and dihedral angles. Under biological conditions, the bond lengths and bond angles are fairly rigid and thus can be assumed to be fixed at their equilibrium values. Using this assumption, the backbone dihedral angles fully determine the geometric shape of the folded protein.

The names of the dihedral angles of a protein chain follow a standard nomenclature. The dihedral angle between the normals of the planes formed by atoms $C'_{i-1}N_iC_{\alpha,i}$ and $N_iC_{\alpha,i}C'_i$ respectively, is called ϕ_i where $i-1$ and i are two adjacent amino acid residues. The angle defined by the planes $N_iC_{\alpha,i}C'_i$ and $C_{\alpha,i}C'_iN_{i+1}$ respectively, is called ψ_i where i and $i+1$ are two adjacent amino acid residues. Also, ω_i is the dihedral angle defined by the planes $C_{\alpha,i}C'_iN_{i+1}$ and $C'_iN_{i+1}C_{\alpha,i+1}$. The letter χ is utilized to denote the dihedral angles which are associated with the side groups R_i. Finally, the letter θ is used to name the dihedral angles associated with the two end groups. Figure 16.1 illustrates these conventions.

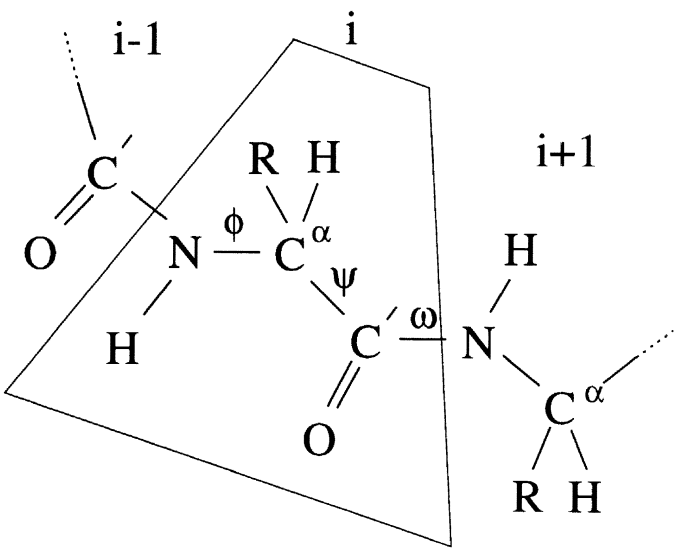


Figure 16.1. Dihedral angles in a protein

16.2.2 Potential Energy Model

The dynamics of atoms in a molecule are governed by the quantum theory of its participating electrons. Using the Born-Oppenheimer approximation, one can determine the energy for fixed atomic nuclei from the smallest eigenvalue of the Hamiltonian of the electron system. These approximations and their derivatives are calculated using *ab–initio* methods. However, due to their computational complexity, such calculations are limited to extremely small molecules.

As a result, many models have been developed using a classical description of molecules in terms of atomic bonds and effective interactions. Some of these parameterizations of molecular potential functions include ECEPP (Momany *et al.*, 1974; Momany *et al.*, 1974; Momany *et al.*, 1975), AMBER (Weiner *et al.*, 1984; Weiner *et al.*, 1986), CHARMM (Brooks *et al.*, 1983a), DISCOVER (Dauber-Osguthorpe *et al.*, 1988), GROMOS (Gunsteren and Berendsen, 1987), MM3 (Allinger *et al.*, 1989), ENCAD (Levitt, 1983), ECEPP/2 (Némethy *et al.*, 1983) and ECEPP/3 (Némethy *et al.*, 1992). In general, these models, also known as force fields, are expressed as summations of empirically derived potential functions, with the mathematical form of individual energy terms based on the phenomenological nature of that term. Constants describing molecular geometry, such as bond lengths and bond angles, are parameterized on empirical structural information. In addition, thermodynamic data from small molecules and spectroscopic data are used to derive the parameters describing the relative strengths of particular interatomic interactions. In most cases, these force fields are atom centered potentials from which the total molecular energy is computed as a sum over all pairwise interactions.

In this chapter, the ECEPP/3 (Empirical Conformational Energy Program for Peptides) potential model is utilized. In this force field, it is assumed that the covalent bond lengths and bond angles are fixed at their equilibrium values. It has been observed that variations in bond lengths and bond angles depend mostly on short range interactions; that is, those between the side chain and backbone of the same residue. Under this assumption, all residues of the same type have essentially the same geometry in various proteins (Momany *et al.*, 1975). Therefore, a chain of any sequence can be generated using the fixed geometry specific to each type of amino acid residue in the sequence.

Based on these approximations, the conformation is only a function of the dihedral angles, and hence ECEPP/3 accounts for energy interaction terms which can be expressed solely in terms of the dihedral angles. The total conformational energy is calculated as the sum of the electrostatic, nonbonded, hydrogen bonded, and torsional contributions. There is also a pseudo–potential for loop closing if the polypeptide contains two or more sulfur–containing residues. The work by (Némethy *et al.*, 1992) includes a revised treatment of prolyl and hydroxyprolyl residues. For each prolyl or hydroxyprolyl residue contained in the polypeptide a fixed internal conformational energy for the pyrolidine ring is added. The main energy contributions (electrostatic, nonbonded, hydrogen bonded) are computed as the sum of terms for each atom pair (i, j) whose interatomic distance is a function of at least one dihedral angle. The contributing

terms to the total potential energy of ECEPP/3 are shown in Figure 16.2, and the development of the appropriate parameters is discussed and reported in ECEPP/3 (Némethy *et al.*, 1992).

$$
\begin{aligned}
E = & \sum_{(i,j)\in ES} 332.0\frac{q_i q_j}{D r_{ij}} && \text{(Electrostatic)} \\
+ & \sum_{(i,j)\in NB} F\frac{A}{r_{ij}^{12}} - \frac{C}{r_{ij}^{6}} && \text{(Nonbonded)} \\
+ & \sum_{(hx)\in HX} F\frac{A'}{r_{hx}^{12}} - \frac{B}{r_{hx}^{10}} && \text{(Hydrogen bonded)} \\
+ & \sum_{k\in TOR} \left(\frac{E_0}{2}\right)(1 \pm \cos n_k\theta_k) && \text{(Torsional)} \\
+ & \sum_{l\in LOOP} B_L \sum_{il=1}^{il=3} (r_{il} - r_{io})^2 && \text{(Cystine Loop-Closing)} \\
+ & \sum_{l\in LOOP} A_L (r_{4l} - r_{4o})^2 && \text{(Cystine Torsional)}
\end{aligned}
$$

Figure 16.2. Potential energy terms in ECEPP/3 force field.

16.2.3 Solvation Models

A complete description of the total energy of a polypeptide must also include its interactions with the solvent. Explicit methods can be used by actually surrounding the polypeptide with solvent molecules and calculating solvent–peptide and solvent–solvent interactions using potentials similar to those previously described. Although these methods are conceptually simple, explicit inclusion of solvent molecules greatly increases the computational time needed to simulate the polypeptide system. Therefore, most simulations of this type are limited to restricted conformational searches. In addition, the interactions between the protein molecule and the surrounding water molecules are not fixed for a given peptide configuration. In reality, a large number of solvent configurations must be considered, and the free energy of hydration can then be calculated by averaging over these configurations. Average values for peptide–solvent interaction energies can also be calculated using a mean–field approximation for the free energy of solvation. In this case the interactions are expressed as an average over all positions of solvent molecules for a given protein configuration. However, limitations in these methods have led to the development of simpler implicit solvation models (Vásquez *et al.*, 1994) .

There are a number of semi-empirical models which can be used to predict hydration effects. The main assumption of these models is that, for each functional group of the peptide, a hydration free energy can be calculated from an averaged free energy of interaction of the group with a layer of solvent known as the hydration shell. In addition, the total free energy of hydration is expressed as a sum of the free energies of hydration for each of the functional groups of the peptide; that is, an additive relationship is assumed.

16.2.3.1 MSEED – Accessible Surface Area Model. Accessible surface area methods assume that the free energy of hydration is proportional to the solvent–accessible surface area of the peptide, as described by the following equation:

$$E_{HYD} = \sum_{i=1}^{N} (A_i)(\sigma_i). \tag{16.1}$$

In Equation 16.1, an additive relationship for N individual functional groups is assumed. (A_i) represents the solvent–accessible surface area for the functional group, and (σ_i) is an empirically derived free energy density parameter.

There are a number of ways to define the surface of a peptide. In developing these surfaces the peptide is represented by a union of spheres, with the radii of the spheres set by the van der Waals radii of the constituent atoms. A spherical test probe is then rolled over these spheres, thereby tracing out a surface. The molecular surface is set by direct contact between the probe sphere and the peptide spheres. In areas where the probe cannot make direct contact, the closest part of the probe is used. The solvent–accessible surface is defined by the surface traced by the center of the probe as the probe rolls over the peptide spheres. These areas depend on the radius of the probe sphere; when this radius is set to zero both the molecular and solvent–accessible surface areas become the van der Waals surface of the peptide. Figure 16.3 illustrates this procedure.

In this chapter, solvent–accessible surface areas are calculated using the MSEED (Perrot *et al.*, 1992) program, which employs algorithms developed by Connolly (Connolly, 1983). MSEED eliminates many unnecessary computations by considering only those convex faces that are on the accessible surface. Rigorous implementation of Connolly's method requires the calculation of interior surface areas, which are ultimately found to be zero. A full description of the MSEED algorithm is given by (Perrot *et al.*, 1992).

There are some limitations to these surface area calculations. Each convex face on the surface is defined by the points of intersections for three spheres and by the set of arcs of intersecting circles for two spheres. These points and arcs define a curvilinear polygon on the surface of one sphere, and only contiguous polygons are included in the accessible surface area of the peptide. However, in searching for surface accessible polygons, MSEED uses points of intersection of three spheres (vertices). Obviously, in the case of two intersecting spheres or a completely free sphere MSEED will not include this solvent–accessible surface area. It will also not search domains connected by only two overlapping

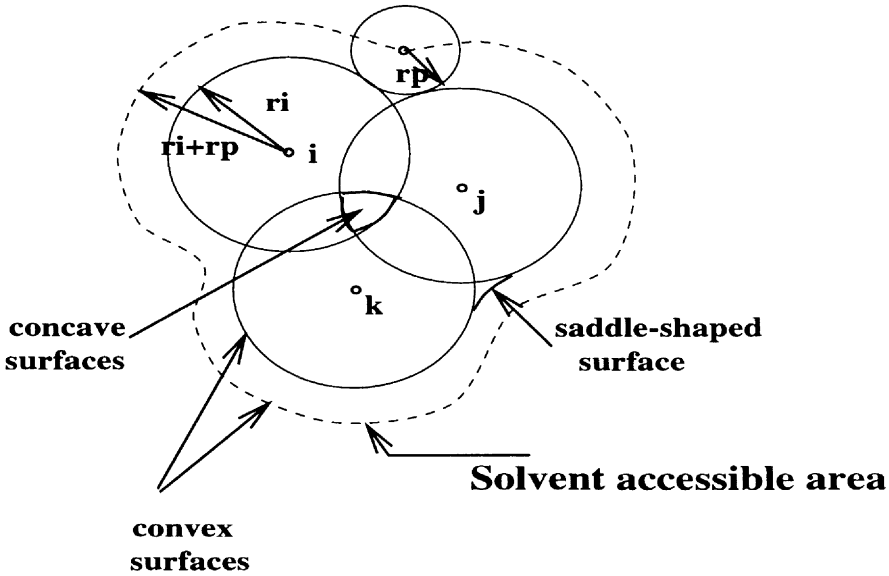


Figure 16.3. Determination of solvent accessible area

spheres. This could be overcome by using multiple starting points to search for other domains. Despite these limitations, it was found that for water, with an effective probe sphere of 1.4 Å, the error in calculating the solvent-accessible area is less than 2 % for a number of test problems (Perrot *et al.*, 1992).

Another problem may occur during minimization of the total energy. Specifically, due to conformational changes during minimization, the area of each atomic surface changes continuously but the gradients may have discontinuities. This occurs when a new vertex or edge appears on the surface. If the discontinuity is large, minimization techniques requiring gradients may fail to converge to the local minimum conformation. A full description of all the situations in which the gradient of the molecular surface area becomes discontinuous has been reported (Wawak *et al.*, 1994).

Once the solvent–accessible surface areas have been calculated, these values must be multiplied by the appropriate (σ_i) parameters as shown in Equation 16.1. There are a number of models available, including JRF, OONS, SR-FOPT, which provide estimates for these parameters based on interactions between water and the functional groups of peptides. It has been shown that minimum energy solvated conformations predicted by the JRF model provided the best correspondence to native (crystallographic) structures when compared with other models (Williams *et al.*, 1992). These parameters were derived from NMR studies of low energy solvated configurations of 13 tetrapeptides. An ensemble of low energy structures for these tetrapeptides was also developed using the ECEPP/2 potential function, and a nonlinear least–squares system

was optimized for the best set of atomic solvation parameters. Because it was developed from minimum energy conformations of peptides, the JRF parameter set has been shown to produce undesirable perturbations during local minimizations if the solvation energy contributions are added at every iteration. Therefore, the surface–accessible solvation energies are only included at local minimum conformations. A detailed comparative study of several solvation models coupled with rigorous global optimization searches can be found in Klepeis and Floudas (1999a).

16.2.3.2 RRIGS – Accessible Volume Shell Model. In accessible volume models, the free energy of hydration is assumed to be proportional to the water–accessible volume of a hydration layer surrounding the peptide. This can be represented in a form very similar to Equation 16.1:

$$E_{HYD} = \sum_{i=1}^{N} (VHS_i)(\delta_i). \tag{16.2}$$

An additive relationship for the N individual atoms of the peptide is assumed, and (VHS_i) represents the solvent–accessible volume of hydration shell for each atom i which is exposed to water. Finally, the (δ_i) are empirically determined free energy of hydration densities for these atoms.

The hydration shell is defined by the volume inside a sphere of radius R_i^h but outside a sphere of radius R_i^v, with both radii centered on atom i. The larger radius, R_i^h, corresponds to the radius of the first hydration shell of atom i, while R_i^v is equal to the van der Waals radius. In order to calculate (VHS_i), the volume of a collection of overlapping hard spheres must be computed using:

$$V(\mathbf{R}) = \sum_i a_i S_i - \sum_{ij} b_{ij} D_{ij} + \sum_{ijk} c_{ijk} T_{ijk} - \sum_{ijkl} d_{ijkl} Q_{ijkl}. \tag{16.3}$$

In Equation 16.3, S_i signifies the volume of a single sphere, while D_{ij}, T_{ijk} and Q_{ijkl} represent the volume of intersection of two, three and four spheres, respectively. This is sufficient because all higher order overlaps can be decomposed into the three types of intersections included in Equation 16.3. Therefore, the solvent–accessible volume of hydration can be written as:

$$(VHS_i) = V(R_i^h) - V(R_i^v). \tag{16.4}$$

The first term in Equation 16.4 is calculated using Equation 16.3 with the radii of all atoms set equal to their van der Waals radii, while the second term is calculated with the radius of atom i equal to R_i^h and the van der Waals radii of all the other atoms. Although the calculation of the (VHS_i) is explicit, the form of Equation 16.3 is not suitable for force field models using pairwise intramolecular potential, such as ECEPP/3. Furthermore, direct truncation at the double–overlap term would lead to large errors.

In this chapter, the RRIGS (Reduced Radius Independent Gaussian Sphere) approximation is used to efficiently calculate the exposed volume of the hydration shell (Augspurger and Scheraga, 1996). This method uses a truncated form of Equation 16.3 but also artificially reduces the van der Waals radii of all atoms other than atom i when calculating (VHS_i). These reductions effectively decrease the contribution of the double overlap terms, leading to a cancellation of the error which results from neglecting the triple and higher overlap terms. In addition, the characteristic density of being inside the overlap volume of two intersecting spheres is not represented as a step function, but as a Gaussian function, which provides continuous derivatives of the hydration potential. Therefore, the solvation energy contributions can easily be added at every step of local minimizations.

The reduced van der Waals radii are fixed and do not need to be recalculated for every configuration because the fixed geometry assumption was used (as in ECEPP/3). The contribution of covalently bonded atoms to (VHS_i) is also fixed because of these assumptions. Therefore, the RRIGS approximation has the same set of interactions as the ECEPP/3 potential. Finally, using the exact (VHS_i) expressions, values for the empirical free energy densities, (δ_i), were developed by a least square fitting of experimental free energy of solvation data for 140 small organic molecules (Augspurger and Scheraga, 1996).

16.3 GLOBAL OPTIMIZATION

The energy minimization problem can be formulated as a nonconvex nonlinear optimization problem. Let $i = 1, \ldots, N_{RES}$ be an indexed set describing the sequence of amino acid residues in the peptide chain. There are ϕ_i, ψ_i, ω_i, $i = 1, \ldots, N_{RES}$ dihedral angles along the backbone of this peptide. In addition, let $k = 1, \ldots, K^i$ denote the dihedral angles of the side chains for the ith residue and $j = 1, \ldots, J^N$ denote the dihedral angles of the amino end group and $j = 1, \ldots, J^C$ of the carboxyl end group, respectively. Therefore, these angles can be defined in following manner : χ_i^k, $i = 1, \ldots, N_{RES}$, $k = 1, \ldots, K^i$ for the side chain dihedral angles; θ_j^N, $j = 1, \ldots, J^N$ and θ_j^C, $j = 1, \ldots, J^C$ for the amino and carboxyl end group dihedral angles, respectively. Using these definitions the optimization problem takes the following form:

$$\begin{aligned}
\min \quad & E(\phi_i, \psi_i, \omega_i, \chi_i^k, \theta_j^N, \theta_j^C) && (16.5)\\
\text{subject to} \quad & -\pi \leq \phi_i \leq \pi, \ i = 1, \ldots, N_{RES} \\
& -\pi \leq \psi_i \leq \pi, \ i = 1, \ldots, N_{RES} \\
& -\pi \leq \omega_i \leq \pi, \ i = 1, \ldots, N_{RES} \\
& -\pi \leq \chi_i^k \leq \pi, \ i = 1, \ldots, N_{RES}, \ k = 1, \ldots, K^i \\
& -\pi \leq \theta_j^N \leq \pi, \ j = 1, \ldots, J_N \\
& -\pi \leq \theta_j^C \leq \pi, \ j = 1, \ldots, J_C.
\end{aligned}$$

Notice that this formulation corresponds to the class of box constrained nonlinear twice differentiable optimization problems that we discussed in Chapter 11 from a global optimization point of view. In general, E represents the total potential energy function and the free energy of solvation. For accessible volume shell hydration (RRIGS) this is the exact formulation. However, in the case of surface–accessible hydration (MSEED), the potential energy function is minimized before adding the hydration energy contributions. This approach is represented by the following equation:

$$E_{Total} = E_{Min}^{Unsol} + E^{Sol}. \tag{16.6}$$

Even after reducing this optimization problem to a function of internal variables (dihedral angles), the multidimensional surface that describes the energy function has a very large number of local minima. In addition, evaluation of the potential, especially with the addition of solvation, is computationally expensive, which makes even local minimization slow.

In this chapter, the αBB global optimization approach is employed in the identification of global minimum energy conformations of solvated peptides. The development of this branch and bound method was motivated by the need for an algorithm that could guarantee convergence to the global minimum of nonlinear optimization problems with twice–differentiable functions (Floudas, 1997). The application of this algorithm to the minimization of potential energy functions was first introduced for microclusters as discussed in Chapter 14 (Maranas and Floudas, 1992; Maranas and Floudas, 1993), and small acyclic molecules as discussed in Chapter 15 (Maranas and Floudas, 1994a; Maranas and Floudas, 1994b). The αBB approach has also been extended to constrained optimization problems (Androulakis *et al.*, 1995; Adjiman and

Floudas, 1996; Adjiman *et al.*, 1998a; Adjiman *et al.*, 1998b). In more recent work, the algorithm has been shown to be successful for isolated peptide systems using the realistic ECEPP/3 potential energy model (Maranas *et al.*, 1996; Androulakis *et al.*, 1997a), as well as for solvated peptides (Klepeis *et al.*, 1998; Klepeis and Floudas, 1999a).

16.3.1 Minimization of Potential Energy using αBB

As we discussed in Chapters 11 and 12, the αBB global optimization algorithm effectively brackets the global minimum solution by developing converging lower and upper bounds. These bounds are refined by iteratively partitioning the initial domain. Upper bounds on the global minimum are obtained by local minimizations of the original energy function, E. Lower bounds belong to the set of solutions of the convex lower bounding functions, which are constructed by augmenting E with the addition of separable quadratic terms. The lower bounding function (L) of the energy hypersurface can be expressed in the following manner:

$$
\begin{aligned}
L \;=\; E + \{ \quad & \sum_{i=1}^{N_{RES}} \alpha_{\phi,i} \left(\phi_i^L - \phi_i\right) \left(\phi_i^U - \phi_i\right) + \\
& \sum_{i=1}^{N_{RES}} \alpha_{\psi,i} \left(\psi_i^L - \psi_i\right) \left(\psi_i^U - \psi_i\right) + \\
& \sum_{i=1}^{N_{RES}} \alpha_{\omega,i} \left(\omega_i^L - \omega_i\right) \left(\omega_i^U - \omega_i\right) + \\
& \sum_{i=1}^{N_{RES}} \sum_{k=1}^{K^i} \alpha_{\chi,i,k} \left(\chi_i^{k,L} - \chi_i^k\right) \left(\chi_i^{k,U} - \chi_i^k\right) + \\
& \sum_{j=1}^{J^N} \alpha_{j,\theta^N} \left(\theta_j^{N,L} - \theta_j^N\right) \left(\theta_j^{N,U} - \theta_j^N\right) + \\
& \sum_{j=1}^{J^C} \alpha_{j,\theta^C} \left(\theta_j^{C,L} - \theta_j^C\right) \left(\theta_j^{C,U} - \theta_j^C\right) \} .
\end{aligned}
\tag{16.7}
$$

Here $\phi_i^L, \psi_i^L, \omega_i^L, \chi_i^{k,L}, \theta_j^{N,L}, \theta_j^{C,L}$ and $\phi_i^U, \psi_i^U, \omega_i^U, \chi_i^{k,U}, \theta_j^{N,U}, \theta_j^{C,U}$ represent lower and upper bounds on the dihedral angles $\phi_i, \psi_i, \omega_i, \chi_i^k, \theta_j^N, \theta_j^C$. The α represent nonnegative parameters which must be greater or equal to the negative one-half of the minimum eigenvalue of the Hessian of E over the defined domain. The overall effect of these terms is to overpower the nonconvexities of the original nonconvex terms by adding the value of 2α to the eigenvalues of the Hessian of E. According to our discussion in Chapters 11 and 12, the convex lower bounding functions, L, possesses a number of important properties which guarantee global convergence (Maranas and Floudas, 1994b):

(i) L is a valid underestimator of E;

(ii) L matches E at all corner points of the box constraints;

(iii) L is convex in the current box constraints;

(iv) the maximum separation between L and E is bounded and proportional to α and to the square of the diagonal of the current box constraints. This property ensures that an ϵ_c convergence tolerance can be reached for a finite size partition element;

(v) the underestimators L constructed over supersets of the current set are always less tight than the underestimator constructed over the current box constraints for every point within the current box constraints.

The computational requirement of the αBB algorithm is proportional to the number of variables (global) on which branching occurs. Therefore, these global variables need to be chosen carefully. Obviously, in a qualitative sense, the branching variables should correspond to those variables which substantially influence the nonconvexity of the surface and the location of the global minimum. With this in mind, (Adjiman *et al.*, 1996; Adjiman *et al.*, 1998b) have developed principles to help identify the important variables, and proposed a variety of rigorous methods for the calculation and updates of the α parameters.

In terms of the protein folding problem, it is generally accepted that the back–bone dihedral angles (ϕ and ψ) are the most influential variables. Therefore, in larger problems involving peptides, the global variable set can include only the ϕ and ψ variables while the dihedral angles associated with the peptide bond (ω) and the side chains (χ) can be treated as local variables.

16.3.2 Algorithmic Description

The determination of the global minimum energy conformation, and thus native conformation of a given peptide using αBB requires the interfacing of several programs. NPSOL (Gill *et al.*, 1986), a local nonlinear optimization solver, and PACK (Scheraga, 1996), a peptide generation program, are directly linked to αBB. PACK internally transforms from Cartesian to internal coordinate systems, and provides function and gradient contributions using ECEPP/3. MSEED (Perrot *et al.*, 1992), the surface–accessible solvation program, is called directly from αBB because its contributions are only required at local minima. On the other hand, RRIGS (Augspurger and Scheraga, 1996), the accessible volume shell solvation model, has been interfaced with PACK and is called through PACK from αBB. These interfaces are shown schematically in Figure 16.4.

The basic steps of the algorithm are as follows:

(STEP 1) The local solver (NPSOL (Gill *et al.*, 1986)) identifies a local minimum of the energy function supplied by PACK in a domain (rectangle)

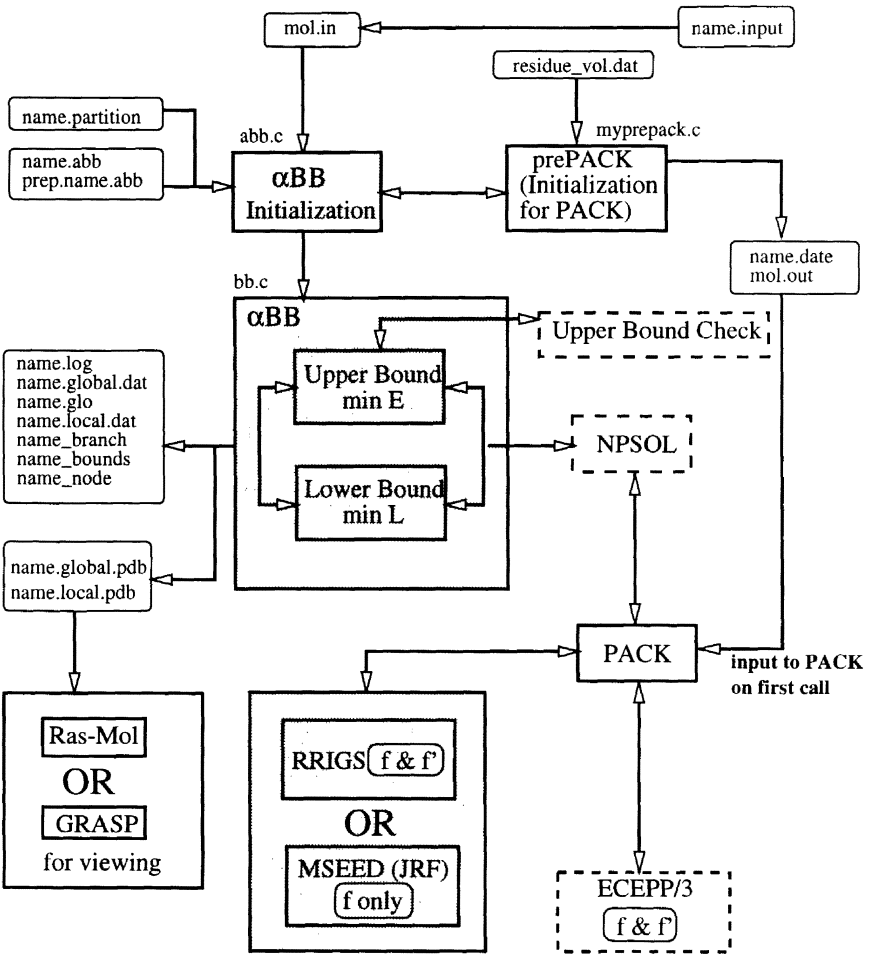


Figure 16.4. Interface for global optimization

defined by the original lower and upper bounds of the variables, with these bounds supplied by αBB. The potential energy evaluation is supplied by ECEPP/3 and, when considering accessible volume shell solvation, the hydration energy is supplied by RRIGS. This is an upper bound on the global minimum solution in the current rectangle.

If surface–accessible solvation effects are desired, the hydration energy is calculated by MSEED at this local minimum. This energy is added to the potential energy to yield a total energy which will serve as an upper bound on the global minimum solution.

(STEP 2) The current best upper bound is updated to be the minimum of those thus far stored.

(STEP 3) The current rectangle is partitioned (e.g., bisection).

(STEP 4) The convex function L is minimized in each rectangle using NPSOL, PACK, ECEPP/3 (and RRIGS). In the case of surface–accessible solvation, the hydration energy is added at the corresponding minima. If a solution is greater than the best upper bound the entire subregion can be fathomed, otherwise the solution will be stored.

(STEP 5) The rectangle with the current minimum value for L is selected for further partitioning.

(STEP 6) If the best upper and lower bounds are within ϵ the program will terminate, otherwise it will return to Step 1.

A detailed description of the algorithmic procedure and its interface with the αBB can be found in Klepeis *et al.* (1997).

16.3.3 Probability–Based Partitioning

In the original problem formulation the dihedral angles were allowed to vary over the entire $[-\pi, \pi]$ domain. It was observed that more intensive computational effort is required for oligopeptides (Androulakis *et al.*, 1997a). Therefore, a reduction of the domain space was proposed based on dihedral angle distributions. Obviously, for the algorithm to be successful these reductions must not exclude the region of the global minimum conformation.

The analysis of ϕ,ψ space of amino acids was first proposed by (Ramachandran and Saisekharan, 1968). Using a hard–sphere potential model, plots of allowable ϕ,ψ space exhibit similar patterns for all naturally occurring amino acids. Similar results were obtained when considering low energy amino acid conformations using the ECEPP/3 force–field (Vásquez *et al.*, 1983). In order to extend this analysis to polypeptide conformations, ϕ,ψ plots were also derived using experimentally obtained conformational data of polypeptides (Lambert and Scheraga, 1989). The results were similar to the original Ramachandran plots, but the plots also identified reduced subdomains which in turn identified specific patterns of polypeptide configurations.

A similar approach was followed when defining reduced subdomains for the initialization of the αBB algorithm (Androulakis *et al.*, 1997a). Specifically, an analysis of 98 proteins from the Brookhaven X–ray data bank provided dihedral angle distributions in the form of histograms from $-\pi$ to π for each dihedral angle of each of the naturally occurring amino acids. Based on these one–dimensional distributions, a number of reduced multi–dimensional domains were identified. In the special case of alanine residues, dihedral angle distributions were also determined for ala–ala and ala–ala–ala sequences. The combination of these sequences defined further reduced domains of the alanine ϕ,ψ space.

	ϕ	ψ	ω	χ_1	χ_2	χ_3	χ_4
tyr	-180,0	-75,50 50,175	160,200	-180,180	-180,180	-180,180	
gly	-180,-30 30,180	-180,0 0,180	160,200				
phe	-180,-50	-75,50 50,175	160,200	-180,180	-180,180		
met	-180,-50	-75,50 50,175	160,200	-180,180	-180,180	-180,180	-180,180
leu	-180,-50	-75,50 50,175	160,200	-180,180	-180,180	-180,180	-180,180

Table 16.1. Bounds on dihedral angles. 1: bounds based on the –ala–ala–ala– pattern, 2: bounds based on the –ala– pattern, 3: The ϕ value for the down puckering of proline that ECEPP/3 uses is -68.8

Based on this procedure, a set of reduced domains can be defined for every dihedral angle of every residue in the oligopeptide sequence. The original optimization problem can be reformulated as:

$$\min \quad E(\phi_i, \psi_i, \omega_i, \chi_i^k, \theta_j^N, \theta_j^C) \tag{16.8}$$

$$\begin{aligned}
\text{subject to} \quad \phi_i &\in \Phi_{i\phi}, \; i_\phi = 1, \ldots, N_{i\phi}, \; i = 1, \ldots, N_{RES} \\
\psi_i &\in \Psi_{i\psi}, \; i_\psi = 1, \ldots, N_{i\psi}, \; i = 1, \ldots, N_{RES} \\
\omega_i &\in \Omega_{i\omega}, \; i_\omega = 1, \ldots, N_{i\omega}, \; i = 1, \ldots, N_{RES} \\
\chi_i^k &\in X_{i\chi}^k, \; i_\chi = 1, \ldots, N_{i\chi}^k, \; i = 1, \ldots, N_{RES}, \; k = 1, \ldots, K^i \\
\theta_j^N &\in \Theta_{j_\theta^N}, \; j_\theta^N = 1, \ldots, N_{\theta N}^j, \; j = 1, \ldots, J_N \\
\theta_j^C &\in \Theta_{j_\theta^C}, \; j_\theta^C = 1, \ldots, N_{\theta C}^j, \; j = 1, \ldots, J_C.
\end{aligned}$$

Here $\Phi_{i\phi}, \Psi_{i\psi}, \Omega_{i\omega}, X_{i\chi}^k$ define the reduced subdomains of each dihedral angle for residue i. The number of subdomains are indicated by $N_{i\phi}, N_{i\psi}, N_{i\omega}, N_{i\chi}^k$. The allowable dihedral angles for each residue must belong to one of the following domains:

$$\begin{aligned}
D_{ji} &= \Phi_{i_\phi} \times \Psi_{i_\psi} \times \Omega_{i_\omega} \times X_{i_\chi}^k \\
ji &= 1, \ldots, N_i \\
N_i &= |\Phi_{i\phi}||\Psi_{i\psi}||\Omega_{i\omega}||X_{i\chi}^k|.
\end{aligned} \tag{16.9}$$

Using these definitions, the total number of initial domains is given by:

$$N = (\prod_{i=1}^{N_{RES}} N_i)|\Theta_{j_\theta^N}||\Theta_{j_\theta^C}|. \tag{16.10}$$

These domains correspond to the Cartesian products of all the sub–domains D_{ji}, $ji = 1, \ldots, N_i$. The reduced domains for the residues in the studied oligopeptides, namely met–enkephalin and leu–enkephalin are defined in Table 16.1. For example, the partitioning for met–enkephalin results in 128 subdomains. Each of these domains is included in the αBB implementation.

16.4 COMPUTATIONAL STUDIES : UNSOLVATED OLIGOPEPTIDES

The αBB coupled with ECEPP/3 has successfully addressed the calculation of the global minimum of unsolvated single–residue and di-peptides. Various computational results were presented by Maranas *et al.* (1996). The proposed approach has also been tested on a number of oligopeptide potential energy examples. The selected relative convergence tolerance is 10^{-2} and the computational requirements (reported in seconds) are on an HP-9000/730 workstation. In the computational studies of unsolvated met-enkephalin and leu-enkephalin the value of the parameter α was set to 3.5.

16.4.1 Unsolvated Met–enkephalin

This example illustrates the application of the αBB global optimization approach on a benchmark molecular conformation problem. It involves the identification of the global minimum total potential energy conformation of the pentapeptide molecule met–enkephalin. Met–enkephalin (H-Tyr-Gly-Gly-Phe-Met-OH), is an endogenous opioid linear penta–peptide found in the human brain, pituitary, and peripheral tissues. Its biological function is related with the endogenous response to pain and a large variety of physiological processes. Met-enkephalin consists of 75 atoms and it involves 24 independent dihedral angles, giving rise to a very complex conformational space involving a plethora of local minima which are estimated on the order of 10^{11} (Li and Scheraga (1987)). Met-enkephalin represents a very challenging conformational study because of (i) the large number of local minima which make the energy hypersurface rather bumpy, (ii) the existence of strong local minima at which most local optimization algorithms terminate, (iii) the location of these strong local minima is not necessarily close to the global minimum even though there are strong local minima within 1.0 kcal/mole, (iv) the very special initialization procedures that need to be devised so as to reach the best solution, (v) the failure of any local optimization method if random initial points are considered, and (vi) the reported failure of local optimization methods to locate the global solution even though the starting point is very close to the global minimum.

The global minimum potential energy conformation of Met–enkephalin is shown in Figure 16.5. The values of the 24 dihedral angles at the global minimum conformation are given in Table 16.6.

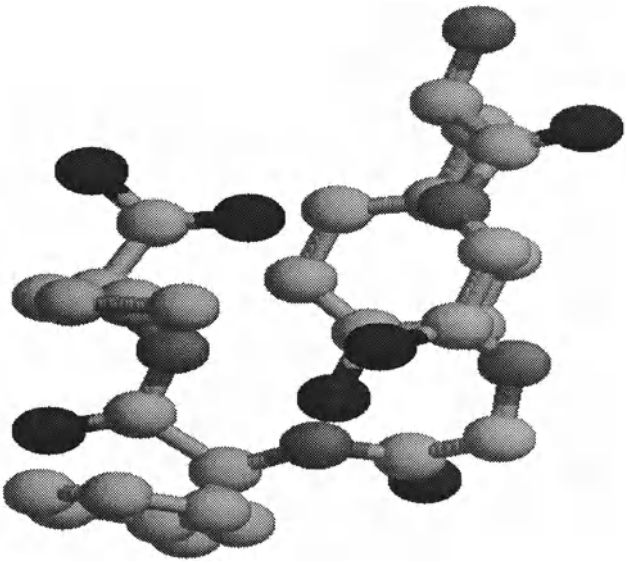


Figure 16.5. Plot of Unsolvated Met–enkephalin conformation, $E^* = -11.707\ kcal/mole$

	ϕ	ψ	ω	χ_1	χ_2	χ_3	χ_4
Tyr	-83.457	155.787	-177.159	-173.205	-100.544	13.636	
Gly	-154.298	86.001	168.507				
Gly	82.964	-75.115	-169.939				
Phe	-136.880	19.079	-174.065	58.843	-85.543		
Met	-163.465	160.944	-179.794	52.884	175.268	-179.851	61.421

Table 16.2. Dihedral angles at the global minimum potential energy conformation of unsolvated met–enkephalin.

As one can observe from Figure 16.5 the global minimum configuration exhibits a bend along the $N-C'$ peptidic bond of Gly^3 and Phe^4, and it represents a type II' β–bend, consistent with observations previously made (Glasser and Scheraga, 1988).

Based on the analysis presented earlier, for all residues the ψ domain is partitioned in two sub–domains and furthermore the ϕ domain of the *glycine* residue is also partitioned into two domains. Table 16.1 presents the corresponding partitioned domains. This results in a partitioning of 128 domains that are considered in the initialization step. Based on the partitioning accord-

ing to the single residue data we define a set of problems, on disjoint domains, that are to be solved. In other words, we consider that the first 7 levels of the branch and bound tree have collapsed into one level, the initial one which is composed of 128 nodes and where each node represents one of these domains. Unlike other approaches, no assumptions are made regarding the values of the χ dihedral angles and they are allowed to vary in the entire interval $[-\pi, \pi]$. In terms of the computational requirements it takes αBB 977 iterations, and 4,669 seconds in order to converge within the required tolerance. It should be noted that the aforementioned CPU time includes the determination of about 100 low energy conformations close to the global minimum of -11.707. At each iteration of the αBB method, the solution of an upper bounding local optimization problem and two convex lower bounding problems are needed. For all three problems the nonlinear solver requires 2-4 CPU seconds per iteration and within this time there are on the average 400 calls to ECEPP/3. At each call to ECEPP/3, the input is a set of values of the dihedral angles. Then, ECEPP/3 transforms the internal coordinates into Cartesian coordinates, performs the function and gradient evaluation in the Cartesian space and transforms them back into the internal coordinates to be used by the local optimization solver. Each call to ECEPP/3 requires 0.005-0.01 CPU seconds, and hence there exists a rather fast calculation of the functions and gradients which in turn require rapid calculation of the nonbonded distances from the internal coordinates.

Met–enkephalin has received a lot of attention and there exists an abundance of computational results that can be used for comparison purposes. Based on the available computational experience the following points can be made:

(i) Comparison of αBB with other approaches

The primary objective in such a comparison of αBB with other procedures is to investigate the computational effort required by the αBB in contrast to the other approaches. Since Met-enkephalin has been studied extensively in the literature with a variety of methods, reported results for several different methods will be presented. Note that due to the difficulties in the search for the global minimum structure of Met-enkephalin there exist a few studies that have treated all 24 dihedral angles as variables, while most of the studies have considered either 10 dihedral angles (i.e., ϕ, and ψ) or 19 dihedral angles (i.e., ϕ, ψ, and χ).

Table 16.3 summarizes the computational requirements for addressing the same problem with a variety of approaches. Based on the above results we can clearly see that αBB compares favorably with most of the widely used methods, and furthermore it provides additional information regarding not only bounds on the global minimum total potential energy, but also on bounds on the values of the dihedral angles.

(ii) Comparison with local optimization methods

A case against using global optimization methods is often made based on the computational requirements and is further suggested that an equivalent

Method	N_{var}	CPU	Computer
Monte Carlo Minimization (Li and Scheraga, 1988)	19 24	2-3 hrs 10 hrs	IBM 3090 IBM 3090
Electrostatically Driven Monte Carlo (Ripoll and Scheraga, 1989)	19 24	2-3 hrs 10 hrs	IBM 3090 IBM 3090
Diffusion Equation (Kostrowicki and Scheraga, 1992)	19	20 min	IBM 3090
Self–Consistent Multitorsional Field (Olszewski *et al.*, 1992)	10	100 min	IBM 3090
Multicanonical Simulated Annealing (Hansmann and Okamoto, 1993)	19	6. hrs	IBM RS6000
Simulated Annealing (Morales *et al.*, 1991)	24	2.5 hrs	Apollo DN1000
Threshold Simulated Annealing (Morales *et al.*, 1992)	24	1.5 hrs	Apollo DN1000
Simulated Annealing with Monte Carlo Minimization (Freyberg and Braun, 1991)	24	2 hrs	CRAY X–MP
Simulated Annealing with Monte Carlo Minimization (Shin and Jhon, 1991)	24	1.2 hrs	CRAY–2S 4 processors
Monte Carlo Minimization vs. Simulated Annealing (Nayeem *et al.*, 1991)	24	1.5-4 hrs	IBM 3090
αBB	24	1.3 hrs	HP–730

Table 16.3. Computational results on Unsolvated Met-enkephalin

amount of local runs would have the same result. Based on the results obtained with αBB, about 1,000 iterations were required. This implies that the non–convex problem was solved about 1,000 times so as to provide upper bounds to our algorithm. In order to test the validity of multiple local runs, we performed over 1,000 local runs from randomly generated starting points and recorded the results. The configuration with the lowest potential energy that is obtained has a value of $E = -8.002\ kcal/mole$ which is over 30% higher in energy than the global minimum potential energy con-

	ϕ	ψ	ω	χ_1	χ_2	χ_3	χ_4
Tyr	102.135	139.309	-166.873	-163.758	79.920	-154.795	
Gly	00.000	60.323	179.446				
Gly	69.574	-85.997	174.502				
Phe	-101.036	-24.937	-172.496	73.661	86.759		
Leu	-81.092	131.465	177.678	-178.432	65.431	-67.792	-180.500

Table 16.4. Dihedral angles for unsolvated Leu–enkephalin of energy $E = -9.332 kcal/mole$.

formation with a potential energy value of $E = -11.707\ kcal/mole$. These results clearly suggest the inadequacy of multistart–like local optimization techniques to address such a complicated problem.

It should also be pointed out that Nayeem *et al.* (1991) provided a direct comparison of the Simulated Annealing, SA, method with the Monte Carlo Minimization, MCM, method for Met-enkephalin. The conclusions drawn from this comparison are : (i) the SA converges to a lower energy structure faster than the MCM method but the SA does not converge to the global minimum in any of the 24 randomly chosen initial conformations whereas the MCM does converge; (ii) the energy difference between the minima reached by the two methods is typically 5-15 kcal/mole in favor of MCM; (iii) even though the SA is followed by local energy minimization, the global optimum solution cannot be reached (i.e., there was an energy decrease of 2.3 kcal/mole but it was higher in energy from the global optimum by 6 kcal/mole), and the RMS deviation was not improved; (iv) even though, instead of using the final conformation reached at the end of a given temperature interval, the minimum energy conformation encountered during that interval was used as the starting point for the next (lower) temperature interval, the global minimum structure could not be reached (i.e., there was a decrease of 2 kcal/mole in energy).

16.4.2 Unsolvated Leu–enkephalin

This peptide is also an endogenous pentapeptide much like met–enkaphalin with the only difference being the fact that *methionine* has been replaced by *leucine*. Figure 16.6 shows the obtained minimum energy conformation found by Androulakis *et al.* (1997a), with a value of α equal to 3.5, in 1027 iterations and 5,209 seconds and a total potential energy of $E = -9.332\ kcal/mole$. The values of the 24 dihedral angles are given in Table 16.10.

Klepeis and Floudas (1999a), as part of their study and implementation of the αBB for solvated peptides, tested the unsolvated Leu-enkephalin with α equal to 5.0 and obtained a slightly lower energy of $-9.349 kcal/mole$. This

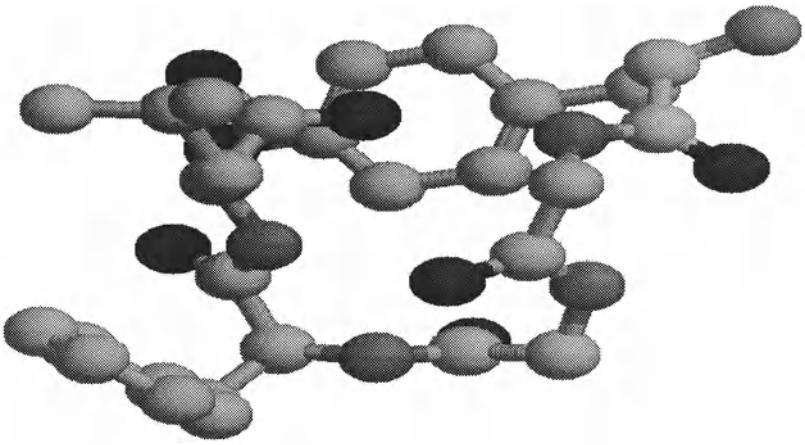


Figure 16.6. Plot of unsolvated leu–enkephalin conformation, $E = -9.332\ kcal/mole$

	ϕ	ψ	ω	χ_1	χ_2	χ_3	χ_4
Tyr	-163.07	-42.29	182.31	-174.79	90.16	-177.26	
Gly	65.90	-88.33	174.19				
Gly	-150.83	31.94	181.29				
Phe	-158.73	157.23	178.02	53.24	84.40		
Leu	-77.83	123.31	181.31	-179.81	64.47	172.58	179.43

Table 16.5. Dihedral angles at the global minimum energy conformation of unsolvated leu–enkephalin. $E = -9.349 kcal/mole$

new unsolvated global minimum structure also possesses a type II' β–bend, although the bend is shifted to the N–C' peptidic bond of Gly^2–Gly^3. The global minimum energy structure is shown in Figure 16.7, with the corresponding dihedral angles given in Table 16.5.

16.5 COMPUTATIONAL STUDIES : SOLVATED OLIGOPEPTIDES

The presented global optimization approach was also applied to a set of solvated 20 uncharged, naturally occurring residues, and several oligopeptides, including met–enkephalin, leu–enkephalin, Ac–Ala_4–Pro–NHMe, and decaglycine. In this section, we discuss the computational studies of the solvated met–enkephalin

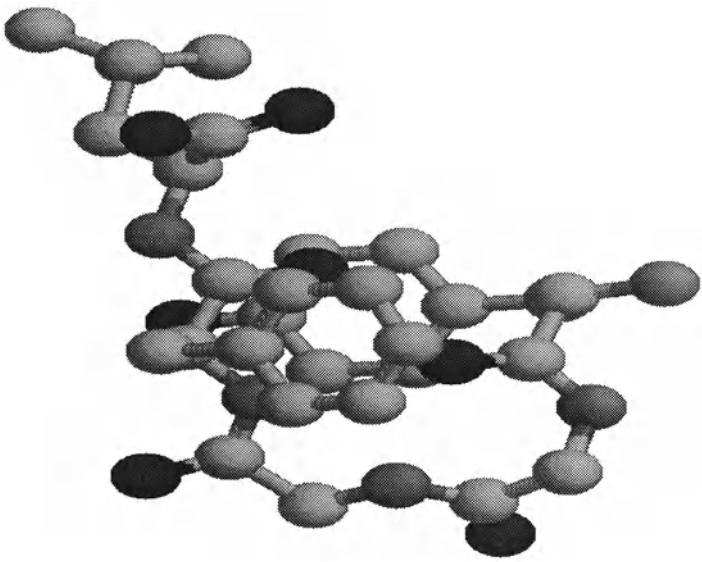


Figure 16.7. Plot of unsolvated leu–enkephalin conformation. Global minimum energy of -9.349 kcal/mole.

and leu–enkephalin. The reader is directed to the Androulakis *et al.* (1997a), Klepeis *et al.* (1998), and Klepeis and Floudas (1999a) for detailed analysis and computational studies of unsolvated and solvated oligopeptides.

16.5.1 Solvated Met–enkephalin

Experimental results have indicated that met–enkephalin in aqueous solution does not possess a unique structure (Graham *et al.*, 1992). In general, the experimentally determined aqueous conformations were found to exhibit characteristics of extended random–coil polypeptides with no discernible secondary structure. When considering the effects of hydration, the competition for backbone hydrogen bonding (with water), which contributes to the bending of the unsolvated conformation, should result in a more extended structure. These qualitative arguments have been confirmed by the analysis of hydrated met–enkephalin using the MSEED model. The plot of the global minimum energy structure, given in Figure 16.8, shows that the residues near the N–terminus are almost fully extended, although there is slight bending near the C–terminus. This bending is stabilized by the formation of a 2.10 $\mathring{A}$ hydrogen bond between the CO of the second glycine residue and the NH proton of the methionine residue. In addition, the structure displays a large 17.00 $\mathring{A}$ separation between the centroids of the Phe and Tyr aromatic rings. The values of dihedral angles corresponding to the global minimum energy of -283.76 kcal/mole are given in

	ϕ	ψ	ω	χ_1	χ_2	χ_3	χ_4
Tyr	-84.96	160.74	179.09	-59.83	100.80	-179.29	
Gly	-160.26	151.83	-177.53				
Gly	159.50	-157.94	178.71				
Phe	-76.55	76.23	-178.05	-61.87	108.68		
Met	-132.90	147.47	-179.83	-65.17	-175.99	-84.91	59.38

Table 16.6. Dihedral angles at the global minimum energy conformation of met–enkephalin, using the MSEED model for hydration.

	ϕ	ψ	ω	χ_1	χ_2	χ_3	χ_4
Tyr	-168.32	-30.81	178.52	-173.58	-101.26	-171.75	
Gly	78.83	-86.96	182.73				
Gly	162.94	91.72	172.83				
Phe	-150.72	162.32	181.50	66.66	92.68		
Met	-77.80	106.79	181.63	-67.82	178.91	180.01	-60.01

Table 16.7. Dihedral angles at the global minimum energy conformation of met–enkephalin, using the RRIGS model for hydration.

Table 16.6. Locating this solution required 1033 iterations and 5,082 seconds (HP-C110).

The RRIGS method predicts a more extended structure than the global minimum structure reported for the unsolvated case (Androulakis *et al.*, 1997a). In fact, although a slight bend occurs near the N–terminus, the structure possesses no hydrogen bonds (< 2.3 Å). In addition, unlike the MSEED structure, there exists close proximity of the Tyr and Phe aromatic rings, as shown in Figure 16.9. The centroids of these rings are separated by 4.16 Å, which is slightly closer than the preferential aromatic–aromatic interaction distance of 4.5 to 7 Å (Burley and Petsko, 1985). Furthermore, the aromatic rings are essentially in a parallel, as opposed to the more common orthogonal, orientation. This suggests an attempt to substantially reduce the hydrophobic exposure of the aromatic side chains. The global minimum conformation, with an energy of -50.01 kcal/mole, was located in 1058 iterations and 8,695 seconds (HP-C110). The values of the dihedral angles are given in Table 16.7.

Additional analysis was performed by calculating (rms) deviation values between both the MSEED and RRIGS global minimum structures and experimentally determined conformations. C^{α} positions for 5 structures of met–enkephalin in aqueous solution (Graham *et al.*, 1992) were generated and compared to both global minima. Average values for the (rms) deviations from the

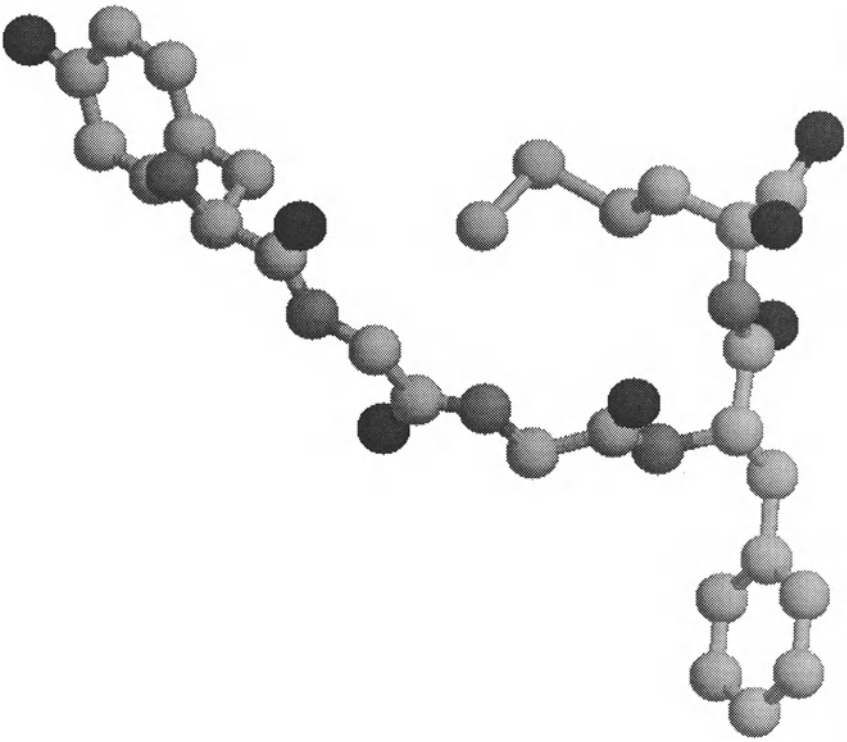


Figure 16.8. Plot of met–enkephalin conformation. Global minimum energy of -283.76 kcal/mole using the MSEED model for hydration. The structure corresponds to the conformational code of FEE*CE. A C^{α} worm is used to highlight the backbone structure.

5 experimentally determined structures were 1.534 and 1.139 for the MSEED and RRIGS global minimum energy structures, respectively. These results suggest that the RRIGS structure may be more "extended". Qualitatively, this agrees with the absence of hydrogen bonding in the RRIGS structure.

It is also interesting to compare energy evaluations at corresponding global minimum solutions. This information is given in Table 16.8. It is apparent that the MSEED model predicts large stabilizing hydration free energies. In addition, these contributions tend to dominate the prediction of the global minimum structure. Specifically, energy evaluations at the RRIGS and unsolvated solutions indicate a substantial increase in overall energy, which can be directly correlated to the increase in hydration free energy. In contrast, this correlation does not hold for the RRIGS model. In fact, the RRIGS model,

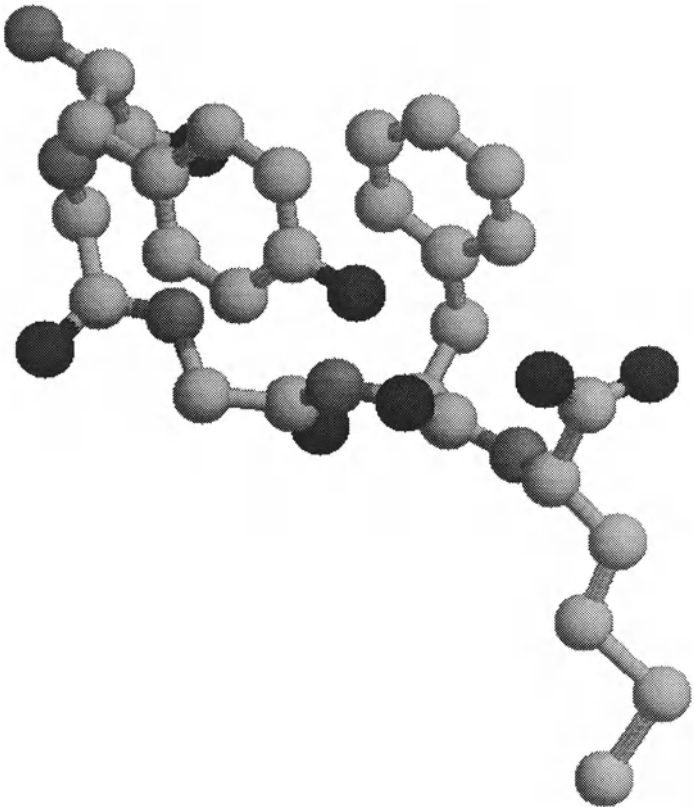


Figure 16.9. Plot of met–enkephalin conformation. Global minimum energy of -50.01 kcal/mole using the RRIGS model for hydration. The structure corresponds to the conformational code of BC*H*EC. A C^{α} worm is used to highlight the backbone structure.

like the MSEED model, predicts a more stabilizing hydration free energy at the MSEED solution. However, nonbonded interactions are less favorable at this solution, resulting in an overall energy increase. In addition, although the solvation free energy becomes less stabilizing at the unsolvated solution, an increase in the number of favorable nonbonded interactions causes the overall energy to be near the global minimum solution.

It should also be noted that the αBB algorithm is able to naturally identify low energy conformers along with the global minimum energy conformation. Table 16.9 lists five local minimum energy conformations within 0.5 kcal of the RRIGS global minimum energy. The structures are related to the global minimum energy conformation as evidenced by their similar conformational

	Global of	E_{TOT}	E_{HYD}	E_{NB}	E_{ES}	E_{TOR}
MSEED	MSEED	-283.77	-288.83	-19.13	23.29	0.90
	RRIGS	-139.35	-130.75	-31.47	21.84	1.03
	UNSOL	-170.88	-159.17	-35.26	21.46	2.09
RRIGS	RRIGS	-50.01	-41.41	-31.47	21.84	1.03
	MSEED	-41.63	-46.69	-19.13	23.29	0.90
	UNSOL	-47.49	-35.78	-35.26	21.46	2.09

Table 16.8. Comparison of hydration energies for met–enkephalin. The first column refers to the hydration model used in the function evaluations, which are performed at the global solutions given in the second column. The total energy, E_{TOT}, is provided along with the contributions from hydration, E_{HYD}, nonbonded interactions (including hydrogen bonding), E_{NB}, electrostatic interactions, E_{ES}, and torsion, E_{TOR}.

Conformer	E_{TOT}	Conformational Code
1	-49.97	BC*G*AG
2	-49.89	BC*H*EG
3	-49.67	BC*H*EB
4	-49.61	BC*H*EA
5	-49.57	BC*GEF

Table 16.9. Low energy conformers (within 0.5 kcal of global minimum energy) for RRIGS model. Total energies and conformational codes are given.

codes (Zimmerman *et al.*, 1977). Such information has important ramifications for more detailed free energy calculations.

16.5.2 Solvated Leu–enkephalin

The inclusion of hydration effects, using the MSEED model, produces an extended solvated conformation. As the plot in Figure 16.10 shows, the residues near the N–terminus are almost fully extended with a slight bend near the C–terminus. This bending is also stabilized by a hydrogen bond; in this case a 2.13 Å hydrogen bond between the CO of the second glycine residue and the NH proton of the leucine residue. The aromatic rings are also widely separated with a distance of 14.44 Å between the two centroids. The values of the 24 dihedral angles, resulting in a global minimum energy of -263.14 kcal/mole, are

	ϕ	ψ	ω	χ_1	χ_2	χ_3	χ_4
Tyr	-84.88	160.00	178.30	-60.54	100.49	-179.21	
Gly	-160.78	140.99	-178.01				
Gly	144.14	-152.83	177.03				
Phe	-79.95	71.30	-176.06	-60.97	108.26		
Leu	-83.99	138.62	-179.24	-53.91	176.56	-178.84	69.81

Table 16.10. Dihedral angles at the global minimum energy conformation of leu-enkephalin, using the MSEED model for hydration.

	ϕ	ψ	ω	χ_1	χ_2	χ_3	χ_4
Tyr	-168.37	-30.66	178.49	-173.40	78.69	-161.13	
Gly	78.92	-87.17	182.69				
Gly	163.20	91.51	172.72				
Phe	-150.66	161.54	181.57	66.75	-86.85		
Leu	-75.45	105.32	181.75	179.5	63.84	172.22	179.31

Table 16.11. Dihedral angles at the global minimum energy conformation of leu-enkephalin, using the RRIGS model for hydration.

given in Table 16.10. This solution was found after 1131 iterations and 5,597 seconds (HP-C110). A plot of this structure is shown in Figure 16.10.

The RRIGS model also predicted a global minimum structure that was more extended than the unsolvated global minimum structure. The backbone structure is essentially identical to the met–enkephalin global minimum energy conformation, with a (rms) deviation of only 0.005. As with met–enkephalin the structure exhibits a bending near the N–terminus with no significant hydrogen bonding (< 2.3 Å). The centroid separation distance of the two aromatic rings is 4.15 Å with a nearly parallel orientation. This conformation, shown in Figure 16.11, corresponds to an energy value of -46.57 kcal/mole, and was found after 1137 iterations and 9,243 seconds (HP-C110). The corresponding values of the dihedral angles are given in Table 16.11.

Experimental evidence suggests that leu–enkephalin conformations also possess extended peptide backbones (Camerman *et al.*, 1983). Using the same set of experimentally determined conformations, average C^{α} (rms) deviations were found to be 1.440 and 1.137 for the MSEED and RRIGS minima, respectively. The average MSEED deviation is slightly better than for met–enkephalin, although the RRIGS deviation remains essentially unchanged because of the similarities of the two RRIGS minimum energy structures. It should also be noted

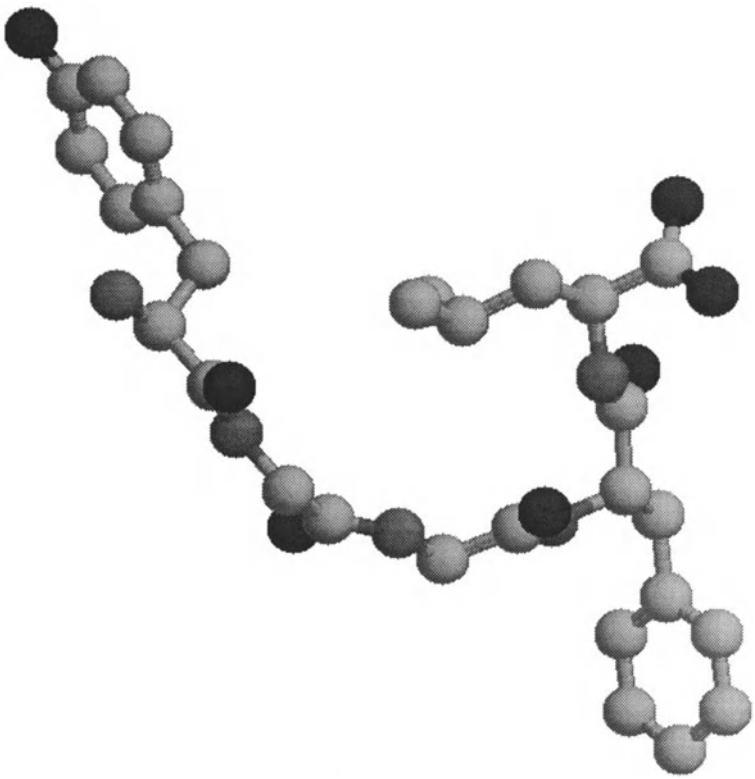


Figure 16.10. Plot of leu–enkephalin conformation. Global minimum energy of -263.14 kcal/mole using the MSEED model for hydration. The structure corresponds to the conformational code of FEE*CF A C^{α} worm is used to highlight the backbone structure.

that the MSEED global minimum structures of the two enkephalins are very similar with a (rms) deviation of only 0.516.

Information on energy evaluations is provided in Table 16.12. The conclusions are qualitatively similar to those made for met–enkephalin. That is, the MSEED model produced hydration free energies which dominated the overall energy. In contrast, although the RRIGS model provided relatively more stabilization at the MSEED global solution, its contribution did not dominate the energy landscape because of the unfavorable nonbonded interactions. Similarly, favorable interactions at the unsolvated global minimum caused its overall energy to be within a few kcal/mole of the global value.

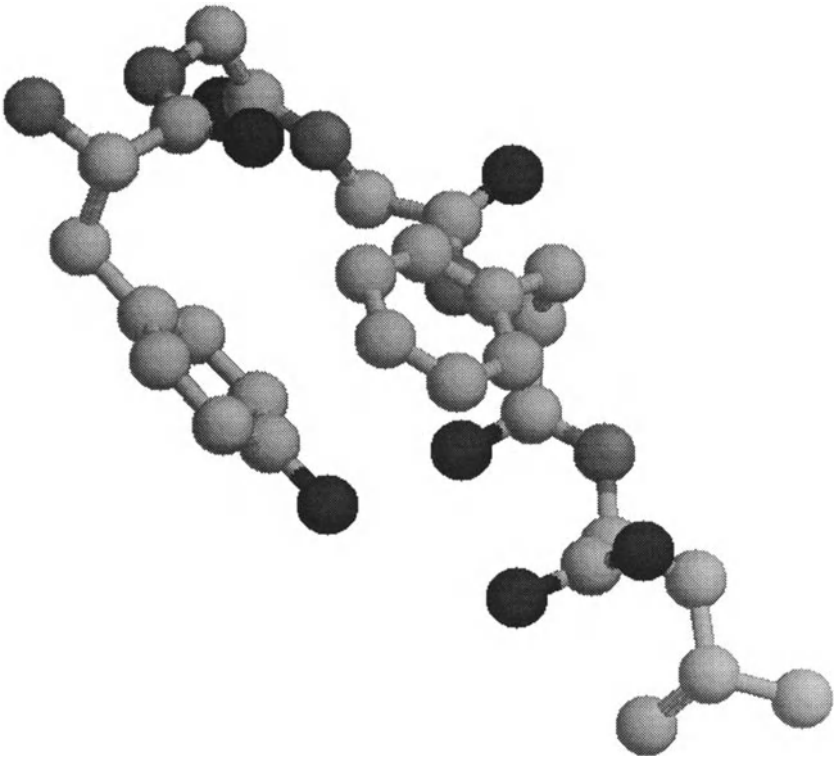


Figure 16.11. Plot of leu–enkephalin conformation. Global minimum energy of -46.57 kcal/mole using the RRIGS model for hydration. The structure corresponds to the conformational code of BC*H*EC A C^{α} worm is used to highlight the backbone structure.

Remark 16.5.1 (Computational Complexity) Based on all computational experience (Maranas *et al.* (1996), Androulakis *et al.* (1997a), Klepeis *et al.* (1998), and Klepeis and Floudas (1999a)) obtained by applying the αBB approach to unsolvated and solvated oligopeptide problems up to thirteen amino acids, the required computational efforts are on the order of n^3.

Summary

A rigorous approach was discussed for identifying the global minimum energy of unsolvated and solvated peptides. The global optimization procedure based on the αBB algorithm is coupled with a novel partitioning of the internal coordinates domain based on probability distributions and it was shown to be

	Global of	E_{TOT}	E_{HYD}	E_{NB}	E_{ES}	E_{TOR}
MSEED	MSEED	-263.14	-268.30	-19.07	23.77	0.46
	RRIGS	-112.60	-105.03	-30.95	22.31	1.07
	UNSOL	-142.84	-133.51	-33.26	20.76	3.17
RRIGS	RRIGS	-46.57	-39.00	-30.95	22.31	1.07
	MSEED	-39.10	-44.26	-19.07	23.77	0.46
	UNSOL	-43.37	-34.04	-33.26	20.76	3.17

Table 16.12. Comparison of hydration energies for leu–enkephalin. The first column refers to the hydration model used in the function evaluations, which are performed at the global solutions given in the second column. The total energy, E_{TOT}, is provided along with the contributions from hydration, E_{HYD}, nonbonded interactions (including hydrogen bonding), E_{NB}, electrostatic interactions, E_{ES}, and torsion, E_{TOR}.

efficient for both unsolvated and solvated implementations. The method was tested using the ECEPP/3 force field, for the unsolvated case, and for two independent solvation models. The MSEED solvation model is based on solvent–accessible surface areas and, using the JRF parameter set, its hydration free energy is added at local minima only. In contrast, implementing the RRIGS solvent–accessible volume of hydration shell model requires function and gradient information at each step of the local minimizations. The pentapeptides of met–enkephalin and leu-enkephalin served as illustrative examples of the presented approach for the determination of their respective global minimum energy conformations. It should be emphasized that the presented global optimization approach provides useful information on (i) the global minimum energy structure, (ii) valid upper and lower bounds on the global minimum structure, (iii) low energy conformers, and (iv) domains within which low energy structures lie.

The presented deterministic global optimization framework for the structure prediction of unsolvated and solvated oligopeptides has been recently employed as a basis for the development of two novel deterministic methods for *Free Energy Calculations* of peptides (Klepeis and Floudas, 1999b).

17 THE αBB APPROACH IN PEPTIDE DOCKING

In this chapter, we discuss the peptide docking problem through the αBB deterministic global optimization framework. Section 17.1 introduces the reader to the peptide docking problem. Section 17.2 presents the various methods that have been proposed for solving the peptide docking problem. Section 17.3 defines the particular peptide docking problem between the HLA-DRB1 and virus peptides. Section 17.4 presents an outline of a novel global optimization approach for the peptide docking problem, while section 17.5 introduces the mathematical modeling and the resulting problem formulation. Section 17.6 presents the global optimization approach αBB, its algorithmic issues and its interface with the atomistic level modeling for the peptide docking problem. Finally, section 17.7 discusses the computational studies for pocket 1 of the HLA-DRB1. The material presented in this chapter is based on the work of Androulakis *et al.* (1997b) and Floudas *et al.* (1999b).

17.1 INTRODUCTION

A natural extension of the protein folding problem is the related problem of peptide docking. In peptide docking problems, equilibrium structures must be identified for a macromolecule-ligand complex. One aspect involves identifying the conformation of the binding site for the "pocket" molecule, which can be accomplished experimentally or approached as an independent protein folding problem. Additionally, equilibrium structures must be identified for a num-

C. A. Floudas, *Deterministic Global Optimization*
© Springer Science+Business Media Dordrecht 2000

ber of candidate "docking" molecules, and a measure of their relative binding affinity must be quantified and compared.

Recent advances in genetic engineering have heightened the interest in research related to predicting docking conformations. The ability to predict these structures is of great theoretical interest, especially in the fields of biophysics and biochemistry. Moreover, the applications of such knowledge also promise to be exciting. For example, the ability to predict these structures would greatly increase our understanding of hereditary and infectious diseases and aid in the interpretation of genome data. In addition to these advances, the ability to understand peptide docking would likely revolutionize the process of de novo drug design.

The complexity associated with the prediction of optimum macromolecular-ligand conformations also complicates the task of modeling these interactions. The first task corresponds to the binding site of the target globular protein that must be correctly characterized, along with the general orientation of the potential ligands. The second task is to select potential binders (ligands), dock these ligands to the active site, and assign a *score* to each complex. These *scores* may then be used to rank binding affinities for a given list of ligands.

The first task usually requires experimental structure determination of the binding site. One particular example that has received much attention has been the major histocompatibility complex (MHC), which plays a major role in immune response (Urban and Chicz, 1996). Crystallographic studies have been performed for the two major classes of MHC molecules, class I (Fremont *et al.*, 1992; Silver *et al.*, 1992) and class II (Stern *et al.*, 1994). This information is invaluable because it can be used to approximate rigid binding sites, which greatly reduces the translational space that must be explored in a conformational search.

The other fundamental feature of the peptide docking problem is the development of accurate scoring functions. Due to the computational complexity of rigorous energy calculations, many methods have relied on qualitative modeling of the peptide docking interactions. In the case of a rigid binding approximation, the use of shape complementarity has had some limited success (Kuntz *et al.*, 1982). These algorithms model the ligand and macromolecule according to their surface topology, and attempt to identify which complexes exhibit the best "fit". These scoring functions are based on the complementarity of the molecules, which, in most cases, is related to their solvent accessible surface areas (Connolly, 1983; Connolly, 1983; Lee and Richards, 1971). The obvious strength of these methods is that they can be made computationally efficient and used to screen large databases of potential ligands. However, studies comparing computational results to experimentally derived, native complexes indicate that many non-native low energy structures are identified. Other methods attempt to refine the scoring function by adding additional criteria, such as conformational energy and solvation energy.

On the other hand, it has been demonstrated that exact modeling of binding free energies provides results in nearly exact quantitative agreement with

experimental results (Grootenhuis and Kollman, 1989; Jorgensen and Tirado-Rives, 1988; Shen and Quiocho, 1995). In contrast to the rigid description of docking, these methods allow for flexibility of both the ligand and receptor molecules. However, for general peptide docking problems, these thermodynamic integration and free energy perturbation methods are computationally infeasible with current computing power. These problems are only tractable when approximate structures are known and relatively small. More detail on these methods can be found elsewhere (Miyamoto and Kollman, 1993; Reynolds *et al.*, 1992). A comprehensive theoretical treatment of the thermodynamics of binding processes in macromolecules is also available (Cera, 1995).

More computationally feasible methods are based on calculating binding free energies using empirically derived free energy functions. Some methods of approximating free energy functions involve structure based potentials (Wallquist *et al.*, 1995). Other approximations utilize parameterization of experimental data to construct scoring functions based on conformational energy, hydrophobic and hydrophilic surface areas, and hydrogen bonding geometries (Head *et al.*, 1996; Verkhivker *et al.*, 1995). However, these methods are generally non-transferable. A general scoring function is represented by :

$$\Delta G = \Delta G_{complex} - \Delta G_{ligand} - \Delta G_{pocket} \tag{17.1}$$

where $\Delta G_{complex}$, ΔG_{binder} and ΔG_{pocket} are the free energies of the peptide docking complex, the free ligand and the free pocket. A more universal approach, applicable to flexible ligands, is to base the free energy calculation on general force field models, which involve potential energy functions similar to those described in Chapter 16. This free energy function must also account for solvation energy, which can be calculated from structure based solvation terms or continuum based models of solvation. Entropic effects of side chain rotations should also be considered. Reviews of methods used to evaluate binding free energies can be found in Jain and Murcko (1995) and Vajda *et al.* (1997).

17.2 PEPTIDE DOCKING METHODS

Once a method for scoring the binding affinity has been selected, the exact form of the optimization problem must be developed. Several general approaches have been employed. A recent review of these methods can be found in Floudas *et al.* (1999b). The most obvious and most difficult approach would be to optimize the entire system of the two interacting peptides. To accomplish this, the relative position of the two peptides, which is defined by six degrees of freedom (three translation and three rotation), along with the total number of internal degrees of freedom for the two molecules must be considered. This problem becomes intractable for all but the smallest systems. Alternative approaches have decomposed the problem by considering the binding affinities of shorter subsequences at different binding sites of the macromolecule (e.g., Androulakis *et al.* (1997b)). The full binding ligand can then be constructed based on the optimally docked subsequences. This approach relies on the ability to build

a suitable ligand. The following discussion classifies the peptide docking approaches according to their treatment of the internal flexibility of the ligand molecule.

Rigid Models. The first, and most common, methods used to address the peptide docking problems are based on the concept of shape complementarity. These methods employ, at least initially, rigid approximations for both the ligand and receptor molecules. In the most general case, six degrees of freedom, three translational and three rotational, must be optimized to determine the best "fit" for the receptor-ligand complex. In practice, approximations are used to effectively reduce the number of degrees of freedom. In addition, the alignment of each ligand must be optimized within the binding site. Typically, several screening stages are used to reduce these optimizations to a manageable number.

Shape based methods include the work of Wodak and Janin (1978), Janin and Wodak (1985), Wodak *et al.* (1987), Lee and Rose (1985), Connolly (1986), and Cherfils *et al.* (1991).

Shape complementarity methods include those of Bacon and Moult (1992), DesJarlais *et al.* (1990), Kuntz *et al.* (1982), DesJarlais *et al.* (1988), Leach and Kuntz (1992), and Shoichet and Kuntz (1991).

Grid search and cubic lattice methods include the contributions of Goodford (1985), Wang (1991), Jiang and Kim (1991), Meng *et al.* (1992), and Pardalos *et al.* (1997).

Flexible Models. In the most general case, flexible docking approaches attempt to optimize the free energy of the entire macromolecule-ligand complex, which is described by translational, rotational and internal variables of the system. In contrast to most rigid modeling approaches, these methods typically do not require prior knowledge of ligand conformations. As a result, their success in predicting ligand binding is highly dependent on the use of detailed scoring functions to evaluate free energy changes. In addition, although some studies have considered full macromolecular-ligand systems, most approaches also depend on effective decomposition strategies of the overall docking problem.

It should also be noted that some simple approaches have been implemented in an attempt to model flexible docking. For example, a number of methods have incorporated ligand flexibility by considering databases of multiple ligand conformations (Kearsley *et al.*, 1994; Miller *et al.*, 1994). However, these methods require reliable databases and methods for developing appropriate ligand conformations, and these are typically not available.

One of the most common approaches has been based on Monte Carlo simulating annealing algorithms by Goodford (1985), and Goodsell and Olson (1990). A second method, also based on simulated annealing, involves a two-step procedure to dock flexible oligopeptide ligands (Calfisch *et al.*, 1992). Another MC based method employs a multiple-start technique in an attempt to reproduce the results of a systematic search (Hart and Read, 1992; Hart and Read, 1994).

Genetic algorithms have recently been used to dock flexible ligands. In some cases, scoring functions have been based on potential energy force fields (Oshiro *et al.*, 1995), although some modified potentials have also been used (Jones *et al.*, 1995). The results of one method (Judson *et al.*, 1995), which includes solvation effects, have emphasized the need for developing reliable scoring functions.

Combinatorial methods have also been used to address the difficulties of modeling full ligand flexibility. In theory, these methods are similar to the build-up methods outlined for the protein folding problem, although in peptide docking intermolecular interactions are also included. Research contributions in this category include the work of DesJarlais *et al.* (1988), Moon and Howe (1991), and Rotstein and Murcko (1993). Similar approaches combine the ideas of fragment assembly and site mapping. In contrast to the single anchor requirement of simple build-up methods, these techniques attempt to identify a number of anchor fragments or residues which can be joined through a process of fragment assembly as studied by Lawrence and Davis (1992), Böhm (1992a), (1992b), Miranker and Karplus (1991), Rosenfeld *et al.* (1993), Calfisch *et al.* (1993), and Gulukota *et al.* (1996).

Finally, a recent novel decomposition based approach that combines atomistic level modeling with rigorous deterministic global optimization has been proposed for predicting peptide docking to the MHC HLA-DR1 protein (Androulakis *et al.*, 1997b).

17.3 PROBLEM DEFINITION

The binding of an influenza virus peptide to the MHC protein HLA-DR1 (Stern *et al.*, 1994) is illustrated in Figure 1. The HLA-DR1 molecule is in gray, while the bound peptide is in white. Notice that the peptide obtains an extended conformation in the binding groove on this complex.

Histocompatibility proteins are organized into two major classes. Polymorphic residues in both class I and II proteins are clustered in the peptide-binding region and are responsible for the different peptide specificities. The major distinctive features of the class I and class II loci are: (i) allograft rejection properties, (ii) relative tissue distributions, and (iii) differing chemical compositions (Paul, 1993). Class I proteins generally bind fragments that range from eight to ten residues in length. Additionally, the protein pockets in this class show allele defined tendencies to bind particular amino acid side chains, or to bind unspecifically. In contrast, class II molecules bind much longer fragments and it has proven difficult to define the binding tendencies of the various pockets (Stern *et al.*, 1994). The different binding properties of these two classes are conjectured to be a result of the more open structure of class II peptides. This allows longer peptides to be situated in the MHC binding groove (Paul, 1993).

In humans, the major histocompatibility proteins are referred to as Human Leukocyte Antigens (HLA). These proteins are then further distinguished by serological studies into HLA-x. Class II loci fall under the classification of

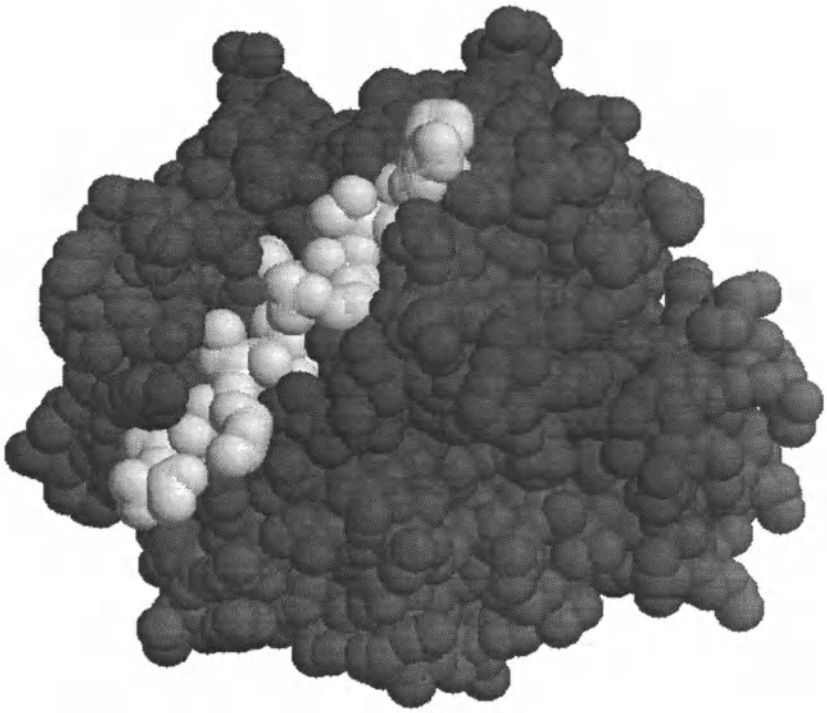


Figure 17.1. HLA-DR1 bound to an influenza virus peptide

HLA-D. The first alleles discovered at this locus were termed HLA-DR (D-related). Subsequently, different class II loci were discovered and now define the categories of HLA-DQ and HLA-DP (Paul, 1993). The HLA-DR1 protein will be the major concern in this Chapter.

The peptide binding site of HLA-DR1 molecule consists of an α chain (33-35 kD), and a β chain (26-28 kD) consisting of 366 amino acid residues. The $\beta 1$ chain of the HLA-DR1 locus is highly variable, while all other regions tend to be relatively invariant.

Crystallographic studies (Stern *et al.*, 1994) have shown that peptide binding is accommodated by five polymorphic pockets on the surface of the HLA-DR1 molecule. Each of these pockets can accommodate a single amino acid residue when a particular peptide is bound (Stern *et al.*, 1994). Accordingly, these pockets play a major role in determining the peptide specificity of class II molecules. Both pocket 1 and pocket 4 have been implicated as playing vital

roles in peptide binding and subsequent recognition by T cells (Hammer *et al.*, 1994; Fu *et al.*, 1995).

Pocket 1 is the largest and deepest pocket of the HLA-DR1 molecule. The area of contact for potential binders has been estimated at 200 $\mathring{A}^2$, (Stern *et al.*, 1994). The pocket has been implicated as being an "anchor" peptide. It has been postulated that the residues that bind to the other four pockets are mainly determined by which residue in the binding peptide attaches to this pocket (Hammer *et al.*, 1994). Pocket 1 consists of hydrophobic residues including several phenylalanine groups. This accounts for the preference of this pocket to accommodate hydrophobic residues, such as tyrosine and phenylalanine (Stern *et al.*, 1994). The large size of this pocket makes it the most solvent accessible of the five pockets.

Pocket 4 is a relatively large pocket that is much shallower than pocket 1. The pocket consists of predominately hydrophobic amino acids, except for a positively charged arginine group. This accounts for this pocket's tendency to bind residues that have large, aliphatic side chains, or negatively charged side chains. Thus, residues such as glutamate and aspartate bind favorably to this pocket while positively charged groups such as lysine, or arginine are repelled (Stern *et al.*, 1994). Pocket 4 has been shown to play an important role in the recognition of the bound peptide by T cells (Fu *et al.*, 1995). Pocket 4, in addition to pockets 6, 7, and 9, has been shown to be 90% inaccessible to solvent (Stern *et al.*, 1994).

The other three pockets (6, 7, and 9) are considered to affect to a lesser degree the determination of peptide binding. Pocket 6 is a shallow pocket that prefers smaller residues. The similarly shallow pocket 7 is nondiscerning in its binding activities, and only partially accommodates side chains. Pocket 9 binds aliphatic side chains due to its small, hydrophobic nature (Stern *et al.*, 1994).

A large array of diseases have been linked to HLA mutations by statistical association. One study has established a definite link between the HLA-DR1 locus and rheumatoid arthritis. It has been shown that a particular peptidic sequence between residues 67 and 71 on the β strand of the HLA molecule plays a significant role the determination of an individual's susceptibility to this disease (Paul, 1993).

Mechanisms for increased disease susceptibility can be posed, but are much more difficult to verify. For example, it is not difficult to postulate that if a mutation weakens antigen binding, then the process of disease identification and eradication will be retarded. But quantitatively establishing binding potential in the laboratory is a difficult task. Thus, diseases have been associated with mutations to various loci within the MHC molecule, but precise mechanisms for disease manifestation have yet to be established. A sample of the diseases associated with the HLA-DR1 locus is present in Table I. The HLA-DR locus that coincides with each of these mutations is also presented. Note that the locus presented here corresponds to DR1(0101). Research work that establishes these connections can be found in Wucherpfennig and Strominger (1995), Hammer *et al.* (1995), Hammer *et al.* (1994) for rheumatoid arthritis, in Wucherpfennig

Changes in Pocket 4 of HLA-DR1	Disease	HLA Locus
β13: Phe $\rightarrow$ Arg β71: Arg $\rightarrow$ Ala	Multiple Sclerosis	DR1(1501)
β13: Phe $\rightarrow$ Arg β71: Arg $\rightarrow$ Ala	Leprosy	DR1(1501)
β13: Phe $\rightarrow$ His β71: Arg $\rightarrow$ Lys	Rheumatoid Arthritis	DR4(0401)
β13: Phe $\rightarrow$ His β70: Gln $\rightarrow$ Asp β71: Arg $\rightarrow$ Glu	Rheumatoid Arthritis	DR4(0402)
β13: Phe $\rightarrow$ His β74: Ala $\rightarrow$ Glu	Rheumatoid Arthritis	DR4(0403)
β13: Phe $\rightarrow$ His	Rheumatoid Arthritis	DR4(0404)

Table 17.1. Mutations to the HLA-DR1 locus and Associated Diseases

et al. (1994), Wucherpfennig and Strominger (1995) for multiple sclerosis, and in Zerva *et al.* (1996) for leprosy.

17.4 OUTLINE OF APPROACH

The modeling and global optimization studies of the interactions between the HLA-DR1 protein and a virus peptide are based on a novel decomposition scheme. As has been described in the previous section, the binding specificity of the HLA-DR1 molecule is mainly determined by the binding characteristics of its five pockets which enables the investigation of each one separately. This chapter focuses on the study of pocket 1. The key ideas in the decomposition approach are: (i) to consider the binding at each pocket separately, (ii) to study the binding of each amino acid to each pocket by considering one at a time, and (iii) to create a rank ordered list of the binding amino acids for each pocket, based on an energetic criterion that reflects the binding specificity.

The decomposition approach consists of the following stages:

Stage 1 In this stage the pocket of the HLA-DR1 protein is represented by a number of residues as described in detail in section 17.5. The work of Stern *et al.* (1994) provides information on the constituent amino acids of each

pocket of HLA-DR1 protein, and provides the Cartesian coordinates of the atoms that participate in each amino acid of the HLA-DR1 protein.

Stage 2 For the specific pocket interacting with each naturally occurring amino acid a mathematical model is formulated that represents all the energetic atom-to-atom interactions. These interactions are classified as (i) *inter*-interactions between the atoms of the residues that define the pocket of HLA-DR1 protein and the atoms of the considered naturally occurring amino acid, and (ii) *intra*-interactions between the atoms of the considered naturally occurring amino acid. These interactions consist of electrostatic, non-bonded, hydrogen-bonding, torsional, and loop-closing components. In addition, *solvation* energy is also considered based on solvent accessible areas. The detailed mathematical model and potential functions used are described in section 17.5.

Stage 3 Having a mathematical model which accounts for all the *inter* and *intra* interactions of the specific pocket and the considered naturally occurred amino acid, in this stage we formulate the global optimization problem which minimizes the total potential energy as it is explained in detail in section 17.5.3.

Stage 4 A deterministic global optimization method, αBB, (Maranas and Floudas, 1994a; Maranas and Floudas, 1994b; Androulakis *et al.*, 1995; Adjiman and Floudas, 1996; Adjiman *et al.*, 1998a; Adjiman *et al.*, 1998b) is employed at this stage for the solution of the resulting nonconvex mathematical model of stage 3. This stage requires the connection of the αBB global optimization method with the conformation energy program PACK (Scheraga, 1996), which utilizes ECEPP/3 (Némethy *et al.*, 1992), and the program MSEED (Perrot *et al.*, 1992), that supplies the solvation contribution as described in detail in section 17.5.

Stage 5 In this stage, we introduce an energy-based criterion that allows for the comparison of the binding between a given pocket and each naturally occurring amino acid. This measure, which is denoted as ΔE, corresponds to the difference of the global minimum total potential energy that is obtained in stage 4 and which is indicated as E_{Total}, and the global minimum potential energy of the considered naturally occurring amino acid when it is far away from the pocket and which is denoted by E^o_{Res}:

$$\Delta E = E_{Total} - E^o_{Res} \tag{17.2}$$

Note here that the energies E_{Total} and E^o_{Res} include the consideration of the solvation energy as it will be discussed in more detail in section 17.5. This criterion represents a measure of the binding affinity of each amino acid to the given pocket, in the sense that it quantifies the tendency of an amino acid to bind with the pocket of the HLA-DR1 molecule. The amino acid that exhibits the least ΔE corresponds to the one with the best possible binding to that pocket of the HLA-DR1 protein.

Stage 6 In this stage we repeat the previous stages for each naturally occurring amino acid and hence create a rank ordered list for the binding of each of them to the specific pocket. The detailed results are shown in section 17.7.

17.5 MATHEMATICAL MODELING AND PROBLEM FORMULATION

17.5.1 Protein Representation

The geometry of a protein can be fully described by defining the relative Cartesian coordinates of each atom. Instead of specifying the coordinate vector for all atoms in a protein, one can specify all bond lengths, covalent bond angles and dihedral angles. Under biological conditions, the bond lengths and bond angles are fairly rigid and thus can be assumed to be fixed at their equilibrium values, and hence the dihedral angles determine the geometric shape of the folded protein. If more than one polypeptide is involved then the relative orientations, and locations of these different chains must be defined. This can most easily be accomplished by defining a translation vector and a rotation matrix. The translation is achieved through the Cartesian coordinates of the initial nitrogen atom of each independent chain. The Euler angles specify the rotations necessary to orient a particular polypeptide and are defined as the angles between the coordinate axes defined by the initial hydrogen, nitrogen, and alpha carbon of each residue. The detailed determination of the Euler angles is given in Androulakis *et al.* (1997b).

17.5.2 Pocket Definition

The relative energies of minimization of each of the five protein pockets are determined mainly by the residues that constitute these pockets. A Program for Pocket Definition, denoted as PPD, constructs these pockets through the selection of all residues that are within a radius $\mathcal{R}$ of the atoms of the crystallographic binder. A range of values for $\mathcal{R}$ have been evaluated in an attempt to discover a radius that realistically represents the pocket, while limiting the number of residues necessary to define the pocket. The information required by the user is provided in a file containing the coordinates of the HLA-DR1 molecule and a file with the coordinates of the influenza virus binding peptide, as well as the value of radius $\mathcal{R}$. Each of the five pockets of the HLA-DR1 molecule were run through PPD for three different radius lengths ($\mathcal{R}$ = 4.0, 4.5, 5.0 Å). The program filters through the α and β strands of the molecule and returns two output files one having a list of all atoms within a radius $\mathcal{R}$, as well as the exact distances, and a PDB file with the coordinates of all residues that define a given protein pocket for the specific radius. Table 17.2 presents the residues defining each of the protein pockets for the various radii considered. There are several important observations that should be made. First, there is the intuitive trend of the pockets becoming more complex with increased radius. Second,

note that pocket 1 is composed of a significantly larger number of residues than any of the other pockets. Finally, note that in some cases an increase in radius does not cause the inclusion of additional amino acids. An example of this is pocket 7, where the pocket is identically defined across the entire range of radii.

The resulting PDB file is then translated to the internal coordinate system by a program denoted as ARAS (Amino acid Residue Angle Solver). The output file obtained by this program is the one required by the conformational energy program PACK to evaluate the potential energy as is described in detail in the next section.

17.5.3 Potential Energy model

The ECEPP/3 (Némethy *et al.*, 1992) detailed potential model is utilized (see also Chapter 16). In this potential model, it is assumed that the covalent bond lengths and angles are fixed at their equilibrium values and the conformational energy is treated as the sum of electrostatic, nonbonded, hydrogen bonding, torsional and cystine contributions.

In addition, the solvation energy is also considered through the utilization of the program MSEED (Perrot *et al.*, 1992) which supplies solvent accessible areas. Once these areas have been calculated, the following formula can be utilized to define the solvation potential:

$$E_{SOL} = \sum_{i=1}^{n} \sigma_{k(i)} \mathcal{A}_i \tag{17.3}$$

where n equals the total number of atoms in the molecule, $\sigma_{k(i)}$ is a coefficient dependent upon the atom type, and $\mathcal{A}_i$ is the solvent accessible area of the ith atom. The σ coefficients were determined by the research performed in Vila *et al.* (1991). The solvent accessible area method was discussed in Chapter 16. Note that E_{SOL} is only added to this overall potential at local minima. This is done because the parameters of the JRF set used in Vila *et al.* (1991) were derived based on a set of tetrapeptide conformations that correspond to local minima of the ECEPP potential energy (Stern *et al.*, 1994). The total energy E_{Total} is then defined as:

$$E_{Total} = E + E_{SOL}. \tag{17.4}$$

17.5.4 Problem Formulation

As it was described earlier, a particular amino acid chain could be defined by a translation vector, a rotation matrix, and the corresponding set of dihedral angles. The translation vector will be defined as the coordinates of the nitrogen atom on the first residue of a chain, while the rotation matrix will be defined by the Euler angles. Since the pocket is considered to be rigid, the only variables

Pocket	$\mathcal{R}$ =	4.0		4.5		5.0
1	ileα31 trpα43 serα53 valβ85 pheβ89	pheα32 alaα52 pheα54 glyβ86	ileα31 trpα43 serα53 valβ85 pheβ89 asnβ82	pheα32 alaα52 pheα54 glyβ86 pheα24	ileα31 trpα43 serα53 valβ85 pheβ89 asnβ82 thrβ90	pheα32 alaα52 pheα54 glyβ86 pheα24 gluα55
4	glnα09 pheβ13 argβ71 tyrβ78	asnα62 glnβ70 alaβ74	glnα09 pheβ13 argβ71 tyrβ78 leuβ26	asnα62 glnβ70 alaβ74 gluα11	glnα09 pheβ13 argβ71 tyrβ78 leuβ26	asnα62 glnβ70 alaβ74 gluα11
6	gluα11 valα65 leuβ11	asnα62 aspα66	gluα11 valα65 leuβ11	asnα62 aspα66	gluα11 valα65 leuβ11 argβ71	asnα62 aspα66 pheβ13
7	valα65 gluβ28 trpβ61 argβ71	asnα69 tyrβ47 leuβ67	valα65 gluβ28 trpβ61 argβ71	asnα69 tyrβ47 leuβ67	valα65 gluβ28 trpβ61 argβ71	asnα69 tyrβ47 leuβ67
9	ileα72 metα73 trpβ09 tyrβ60	asnα69 argα76 aspβ57	ileα72 metα73 trpβ09 tyrβ60	asnα69 argα76 aspβ57 trpβ61	ileα72 metα73 trpβ09 tyrβ60 leuα70	asnα69 argα76 aspβ57 trpβ61

Table 17.2. PPD Pocket Compositions for $\mathcal{R}$ = 4.0-5.0 Å

will be the nitrogen coordinates, Euler angles, and dihedral angles of the amino acid binder.

Let $k = 1, \ldots, K$, where K is the total number of side chain angles of the amino acid residue that attempts to bind the pocket. Then, the set of variable dihedral angles would include the backbone angles (ϕ, ψ, and ω), and the side chain angles (χ^k). The cartesian coordinates of the amino translation vector will be defined by the variables N_x, N_y, and N_z. Similarly, the Cartesian coordinates of the backbone carboxyl carbon are represented by C'_x, C'_y, and

C'_z. Finally, the Euler angles will be represented by ε_1, ε_2, and ε_3. Based on the above definitions the potential energy minimization problem can be formulated as follows:

$$
\begin{array}{lrcl}
\min & & E\left(\phi,\psi,\omega,\chi^k,N_x,N_y,N_z,\varepsilon_1,\varepsilon_2,\varepsilon_3\right) & \\
\text{s.t.} & -\pi & \leq \phi \leq \pi & \\
 & -\pi & \leq \psi \leq \pi & \\
 & -\pi & \leq \omega \leq \pi & \\
 & -\pi & \leq \chi^k \leq \pi, \quad k=1,\ldots,K & \\
 & -\pi & \leq \varepsilon_1 \leq \pi & \\
 & -\pi & \leq \varepsilon_2 \leq \pi & \\
 & -\pi & \leq \varepsilon_3 \leq \pi & \\
 & N_x^l & \leq N_x \leq N_x^u & \\
 & N_y^l & \leq N_y \leq N_y^u & \\
 & N_z^l & \leq N_z \leq N_z^u & \\
 & C_x'^l & \leq C'_x\left(\phi,\psi,\omega,\chi^k,N_x,N_y,N_z,\varepsilon_1,\varepsilon_2,\varepsilon_3\right) \leq C_x'^u & \\
 & C_y'^l & \leq C'_y\left(\phi,\psi,\omega,\chi^k,N_x,N_y,N_z,\varepsilon_1,\varepsilon_2,\varepsilon_3\right) \leq C_y'^u & \\
 & C_z'^l & \leq C'_z\left(\phi,\psi,\omega,\chi^k,N_x,N_y,N_z,\varepsilon_1,\varepsilon_2,\varepsilon_3\right) \leq C_z'^u. &
\end{array}
\tag{17.5}
$$

Note that the superscripts u and l denote upper and lower bounds, respectively, for the Cartesian coordinates of both the amino nitrogen and the carboxyl carbon. In addition to the constraints on the amino nitrogen, there are constraints on the carboxyl carbon. It has been assumed due to the decomposition employed (that enables the consideration of each pocket separately), that the conformational movements of the binding peptide are only constrained by the locations of these two atoms. In the original problem though, the binding residue is part of a longer antigen peptide. The rest of the binding peptide is assumed to bind normally so even if the binding residue is changed these backbone atoms will be relatively confined to their initial positions due to their peptidic linkages. Although the constraints of amino nitrogen can be directly considered in the above formulation since they correspond to problem variables, the C' coordinates are not explicit variables and consequently they must be defined as a function of the other variables (see section 17.6). Note that E is a nonconvex function involving numerous local minima that correspond to metastable states of the specific amino acid binding to the pocket 1.

17.6 GLOBAL OPTIMIZATION APPROACH AND ITS INTERFACE

The global optimization scheme αBB (Maranas and Floudas, 1994a; Maranas and Floudas, 1994b; Androulakis *et al.*, 1995; Adjiman and Floudas, 1996; Adjiman *et al.*, 1998a; Adjiman *et al.*, 1998b) is a deterministic branch and bound algorithm for locating the global optimum based on the construction of converging lower and upper bounds (see Chapters 11 and 12). Upper bounds can be simply obtained by minimizing E using local methods. Lower bounds can be evaluated by constructing the convex underestimator, L, of the original function E and evaluating the single global minimum of the resulting convex problem.

A convex lower bounding function L of potential energy function E can be defined by augmenting E using the ideas of the approach introduced in (Maranas and Floudas, 1994a; Maranas and Floudas, 1994b):

$$
\begin{aligned}
L \;=\; E + \alpha \{ \quad & (\phi^l - \phi)\,(\phi^u - \phi) + \\
& (\psi^l - \psi)\,(\psi^u - \psi) + \\
& (\omega^l - \omega)\,(\omega^u - \omega) + \\
& \sum_{k=1}^{K} (\chi^{k,l} - \chi^k)\,(\chi^{k,u} - \chi^k) + \\
& (N_x^l - N_x)\,(N_x^u - N_x) + \\
& (N_y^l - N_y)\,(N_y^u - N_y) + \\
& (N_z^l - N_z)\,(N_z^u - N_z) + \\
& (\varepsilon_1^l - \varepsilon_1)\,(\varepsilon_1^u - \varepsilon_1) + \\
& (\varepsilon_2^l - \varepsilon_2)\,(\varepsilon_2^u - \varepsilon_2) + \\
& (\varepsilon_3^l - \varepsilon_3)\,(\varepsilon_3^u - \varepsilon_3) \;\}
\end{aligned}
$$

where α is a nonnegative parameter which must be greater or equal to the negative one half of the minimum eigenvalue of the Hessian of E over the rectangular under consideration described by the lower and upper bounds of the involved variables. The properties of function L described in Chapters 11 and 12 enable the construction of a global optimization algorithm. The bounds are successively refined by iteratively partitioning the initial feasible region into smaller ones. The feasible region partition is achieved by subdivision of a rectangle into two subrectangles by halving along the longest side of the initial rectangle (bisection). The convergence proof to an ϵ-global solution in finite steps is discussed in Chapters 11 and 12, while methods for rigorous calculation of the α parameters are discussed in Chapter 12.

17.6.1 Algorithmic Procedure

The proposed approach for the determination of the global minimum of E that corresponds to the peptide conformation binding to the pocket 1 of HLA-DR1 as posed in section 17.3, necessitates the development of an optimization interface that combines the global optimization program αBB, the conformational energy program PACK which utilizes ECEPP/3, the solvation program MSEED, and the local optimization solver NPSOL. Additional program files serve to link these programs. A schematic diagram of the interface between these programs and αBB is shown in Figure 17.2.

The following steps are required in the calculation of the global minimum:

(STEP 1) The local solver NPSOL (Gill *et al.*, 1986) obtains a local minimum of the potential function supplied by PACK in a domain (rectangle) defined by the lower and upper bounds of the variables (bounds are supplied by αBB). PACK determines the energies of individual chains through repeated calls to ECEPP/3.

(STEP 2) The solvation energy at this local minimum is calculated by MSEED. This hydration energy is added to the potential function to yield E_{Total}, which will serve as an upper bound on the global minimum solution in the current rectangle.

(STEP 3) The current best upper bound is updated to be the minimum of those thus far stored.

(STEP 4) The current rectangle is partitioned by bisection along the longest side.

(STEP 5) The convex function L is minimized in each rectangle and the solvation energy is added at the minimum. If a solution is greater than the best upper bound it will be eliminated, otherwise it will be kept on the stack.

(STEP 6) The rectangle with the current minimum solution for L is selected for further partitioning.

(STEP 7) If the best upper and lower bounds are within ϵ the program will terminate, otherwise it will proceed to Step 1.

Remark 17.6.1 It should be noted that ECEPP/3 calculates the potential energy function for a polypeptide chain (Némethy *et al.*, 1992), while PACK is a program that inputs multiple chain data and makes the appropriate calls to ECEPP/3 for calculation of the interaction energies (Scheraga, 1996). Special type penalty functions had to be added to the upper bound function, E, in order to implement the previously discussed constraints on C'. The modified objective function takes the following form:

$$\begin{aligned} E' \;=\; E + \beta\{ \;& \langle C'^{l}_{x} - C'_{x}\rangle + \langle C'_{x} - C'^{u}_{x}\rangle + \\ & \langle C'^{l}_{y} - C'_{y}\rangle + \langle C'_{y} - C'^{u}_{y}\rangle + \\ & \langle C'^{l}_{z} - C'_{z}\rangle + \langle C'_{z} - C'^{u}_{z}\rangle \;\}. \end{aligned}$$

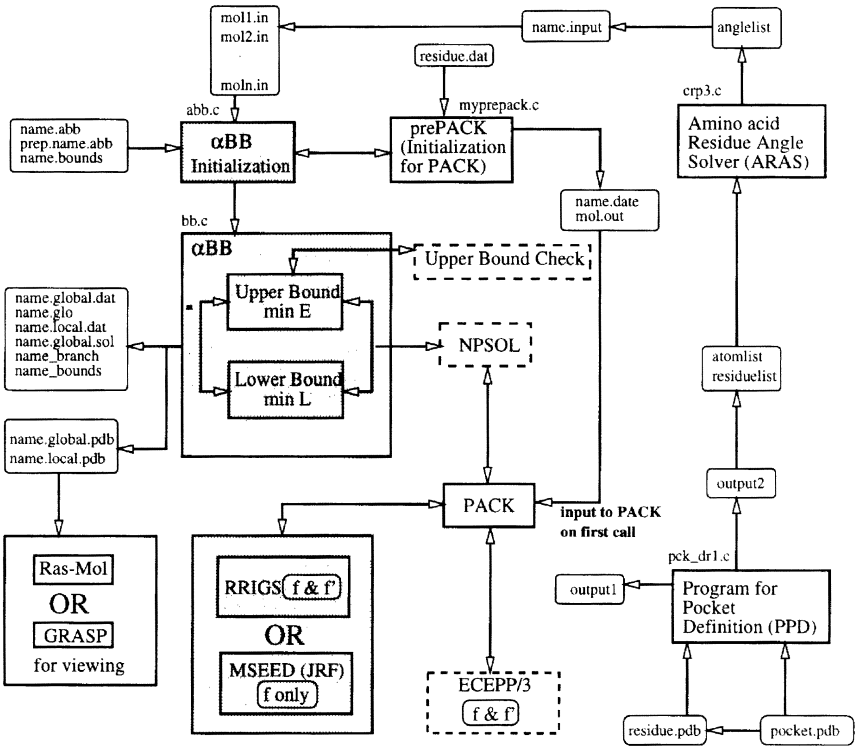


Figure 17.2. The peptide docking interface for global optimization.

The $\langle\rangle$ function is defined as follows: $\langle\mathcal{A}\rangle$ equals $\mathcal{A}$ if $\mathcal{A}$ is greater than zero, otherwise $\langle\mathcal{A}\rangle$ equals zero. Thus, as long as the coordinates are within the defined bounds the objective function will not be modified. Yet, if a particular coordinate falls outside of the bounds, the function will be increased by the value of the transgression multiplied by the arbitrarily large constant β.

Remark 17.6.2 The coordinates of the backbone carboxyl carbon have to be expressed as a function of other variables since they do not correspond to explicit optimization variables. This can be accomplished using analytical geometry and linear algebra. In particular, for each amino acid the coordinates of C' atom can be expressed as a function of the coordinates of the N atom (i.e., the translation vector) denoted as n_x, n_y, n_z, the Euler angles $\theta_1, \theta_2, \theta_3$ and the ϕ dihedral angle, and requires the knowledge of the following parameters : the bond lengths NC^α and C^αC', as well as the value of the angle NC^αC' taken from the literature. The following expressions are derived for the coordinates

of C' atom:

$$
\begin{aligned}
x &= n_x + sc \times temp \times \sin(\theta_1) \times \sin(\theta_2) \\
&+ sa \times sc \times \cos(\phi) \times temp \times \frac{(-\cos(\theta_2) \times \cos(\theta_3) \times \sin(\theta_1) - \cos(\theta_1) \times \sin(\theta_3))}{\sqrt{(1-\cos^2(\phi))}} \\
&+ ((\mathbf{NC}^\alpha) + (\mathbf{C}^\alpha\mathbf{C}') \times \cos(\theta)) \times (\cos(\theta_1) \times \cos(\theta_3) \\
&- \cos(\theta_2) \times \sin(\theta_1) \times \sin(\theta_3))
\end{aligned}
$$

$$
\begin{aligned}
y &= n_y - sc \times temp \times \cos(\theta_1) \times \sin(\theta_2) \\
&+ sa \times sc \times \cos(\phi) \times temp \times \frac{(\cos(\theta_1) \times \cos(\theta_2) \times \cos(\theta_3) - \sin(\theta_1) \times \sin(\theta_3))}{\sqrt{(1-\cos^2(\phi))}} \\
&+ ((\mathbf{NC}^\alpha) + (\mathbf{C}^\alpha\mathbf{C}') \times \cos(\theta)) \times (\sin(\theta_1) \times \cos(\theta_3) \\
&+ \cos(\theta_1) \times \cos(\theta_2) \times \sin(\theta_3))
\end{aligned}
$$

$$
\begin{aligned}
z &= n_z + sc \times temp \times \cos(\theta_2) \\
&+ sa \times sc \times \cos(\phi) \times temp \times \cos(\theta_3) \times \frac{\sin(\theta_3)}{\sqrt{(1-\cos^2(\phi))}} \\
&+ ((\mathbf{NC}^\alpha) + (\mathbf{C}^\alpha\mathbf{C}') \times \cos(\theta)) \times (\sin(\theta_2) \times \sin(\theta_3))
\end{aligned}
$$

where:

$$temp = \sqrt{\frac{(\mathbf{C}^\alpha\mathbf{C}')^2 - (\mathbf{C}^\alpha\mathbf{C}')^2 \times \cos^2(\theta)}{1 + \frac{\cos^2(\phi)}{1-\cos^2(\phi)}}}$$

$$sa = 1.0, sc = -1.0 \text{ if } \phi > 0$$

$$sa = -1.0, sc = 1.0 \text{ if } \phi \leq 0$$

and θ is the angle NC^αC'.

Since the pocket is assumed rigid, the optimization variables are the dihedral angles, the translation vector and the Euler angles of the amino acid under consideration. These variables are partitioned into three sets. The first one (i.e., global variables) consists of the variables where branching occurs; the second set (i.e., local variables) consists of the variables where branching is not performed but they are optimized locally, and the third set (i.e., fixed variables) includes the variables for which there exists sufficient experimental evidence for keeping them fixed.

The information required by the user is provided in three different parts. The first one is required by PACK and contains information about the different protein chains to be considered. The second one is needed by ECEPP/3

and contains information about the sequence and number of the amino acid residues and the type of end groups. It also initializes the dihedral angles, translation vector and Euler angles. The third part provides the bounds on the amino nitrogen and carboxyl carbon of the binding residue. These bounds are defined around the Cartesian coordinates of these two atoms in the influenza virus peptide (Stern *et al.*, 1994). Thus, for each pocket bounds were set in the x, y, z directions around the coordinates of the corresponding atoms of the influenza virus peptide presented by Stern *et al.* (1994). These bounds are given in Table 17.3.

Bounds		*Lower*	*Upper*
N	x	-9.2	-8.4
	y	23.3	24.1
	z	17.3	18.1
C'	x	-10.0	-9.0
	y	25.0	27.0
	z	16.0	18.0

Table 17.3. Bounds on N and C' for pockets 1 of HLA-DRB1

17.7 COMPUTATIONAL STUDIES AND DISCUSSION

In this section, each one of the naturally occurring amino acids will be examined and accessed regarding its binding affinity with Pocket 1 of the HLA-DR1 molecule. The aliphatic side chains of the amino acids Ala, Val, Ile and Leu can be clearly considered as nonpolar, whereas at the opposite end of the polarity scale are the charged residues Glu, Asp, Arg, Lys. Asn and Gly which have amide side chains, as well as the hydroxylic amino acids Ser and Thr are polar and expected to interact strongly with water and have high solubility. The polarity of the rest of the amino acids is more ambiguous. Cys and His have pK_a values close to 7 and may actually be charged in many proteins under physiological conditions. In the computational studies we consider as positively charged the residues Arg+, His+, Lys+ and as negatively charged the residues Asp- and Glu-, using the parameters of ECEPP/3 for eveluating their energy contributions. In Tyr the aromatic ring compensates for the hydroxyl and makes Tyr a nonpolar residue. Gly and Pro are special but from the way they behave in proteins, can be considered as nonpolar and polar, respectively.

17.7.1 Individual Solvated Residues away from Pocket 1

To evaluate the energy of interaction of the 20 naturally occurred amino acids within a pocket the intramolecular energy due to atomic interactions between the atoms of the single residue far away from the pocket has to be calculated. Thus, the global minimum energy for each residue in isolation is found with the consideration of the solvation contribution as described in section 17.4. The results obtained by applying the global optimization algorithm, αBB, (Androulakis *et al.*, 1997b) are shown in Table 17.4, where based on the previous remark regarding amino acid polarity, some are considered to be charged.

17.7.2 Complex of Pocket 1 and Binding Amino Acids

As mentioned earlier, ΔE, defined as the difference $E_{Total} - E^{o}_{Res}$, has been considered to represent the binding potential of a specific naturally occurring amino acid to the pocket considered. As shown in Table 17.2, the number of amino acids included in pocket 1 increase as R increases. Specifically from 4.5 to 5.0 Å the amino acids that were added include threonine and glutamic acid. Note that Glu is negatively charged and is an important factor for evaluating the interactions with positively charged residues, as illustrated in Table 17.6.

The results for pocket 1 with R=5.0 Å are presented in Tables 17.5, 17.6 and in Figure 17.3. Tyr, Phe and Trp are found to have the strongest binding affinities with interaction energies in a range of -20.00 to -16.95. At lower positions in the middle of the list are the Leu, Ile and Val residues having interaction energies between -12.481 and -11.209. At the bottom of the list are the negatively charged residues Glu- and Asp-, which possess 40% smaller interaction energy than that of Val. An interesting result of the theoretical studies is that the positively charged residues appear to be the most unfavorable binders for this pocket.

A series of competitive binding assays was performed that involved analogs of the HA peptide (306-318) and the DRB1*0101 molecule (Monos *et al.*, 1997). Since the HA(306-318) peptide residue that interacts with Pocket 1 is Y(308) a number of analog peptides were synthesized that substituted the Y(308) with 11 different amino acids. The relative binding affinity was derived from the reciprocal of 50% inhibitory concentration (IC50) of each analog peptide in a logarithmic scale. Figures 17.3 and 17.4 show the results of both the theoretical and the experimental results, respectively.

Based on the competitive binding assays shown in Figure 17.4, three groups of binding affinities have been identified. The first group includes the amino acids Tyr, Phe, and Trp which are the residues with the highest affinity to DR1. The second group includes the amino acids Ile, Leu, and Val and are characterized by an intermediate level of affinity to DR1. The third group finally consists of low level affinity amino acids. This group includes the charged

Residue	Code	$E^{o,S}_{Res}$ (kcal/mol)
Ala	A	-42.143
Asn	N	-94.376
Cys	C	-74.667
Gln	Q	-86.964
Gly	G	-56.260
Ile	I	-17.074
Leu	L	-23.166
Met	M	-46.269
Phe	F	-160.850
Ser	S	-82.476
Thr	T	-69.423
Trp	W	-184.230
Tyr	Y	-178.950
Val	V	-25.055
Glu-	E-	-56.607
Asp-	D-	-67.416
His+	H+	-125.460
Arg+	R+	-105.800
Lys+	K+	-27.706

Table 17.4. Standard Energies for Individual Solvated Residues

residues Asp-, Glu-, Arg+, His+ as well as the amino acids Ser and Thr.

Based on the theoretical predictions, shown in Figure 17.3, Tyr, Phe and Trp are at the top positions of the rank ordered list : a result that is further supported from the strong preference of this pocket for large hydrophobic side chains. Furthermore, the amino acids Leu, Ile, and Val were found by the optimization studies to be characterized by potential energies that correspond to 8th, 9th and 12th position on the ordered list, respectively. The binding assays also resulted in intermediate affinities for these amino acids.

Residue	E^{S}_{Total} (kcal/mol)	$E^{o,S}_{Res}$ (kcal/mol)	Δ E (kcal/mol)
Tyr	-198.950	-178.950	-20.000
Phe	-180.475	-160.850	-19.625
Trp	-201.180	-184.230	-16.950
Gln	-102.360	-86.964	-15.396
Met	-60.212	-46.269	-13.943
Asn	-108.160	-94.376	-13.784
Thr	-82.718	-69.423	-13.297
Leu	-35.647	-23.166	-12.481
Ile	-29.539	-17.074	-12.465
Ser	-94.033	-82.476	-11.557
Cys	-85.947	-74.667	-11.280
Val	-36.264	-25.055	-11.209
Ala	-52.498	-42.143	-10.355
Gly	-66.351	-56.260	-10.091
Glu-	-64.531	-56.607	-7.744
Asp-	-69.847	-67.416	-2.431

Table 17.5. Relative Energies for Solvated Residues in Pocket 1 ($\mathcal{R}$ = 5.0Å)

Residue	E^{S}_{Total} (kcal/mol)	$E^{o,S}_{Res}$ (kcal/mol)	Δ E (kcal/mol)
His+	-58.374	-124.870	+66.496
Lys+	+196.15	-27.706	+223.856
Arg+	+182.78	-105.640	+288.420

Table 17.6. Relative Energies for Solvated Positive Charged Residues in Pocket 1 ($\mathcal{R}$ = 5.0Å)

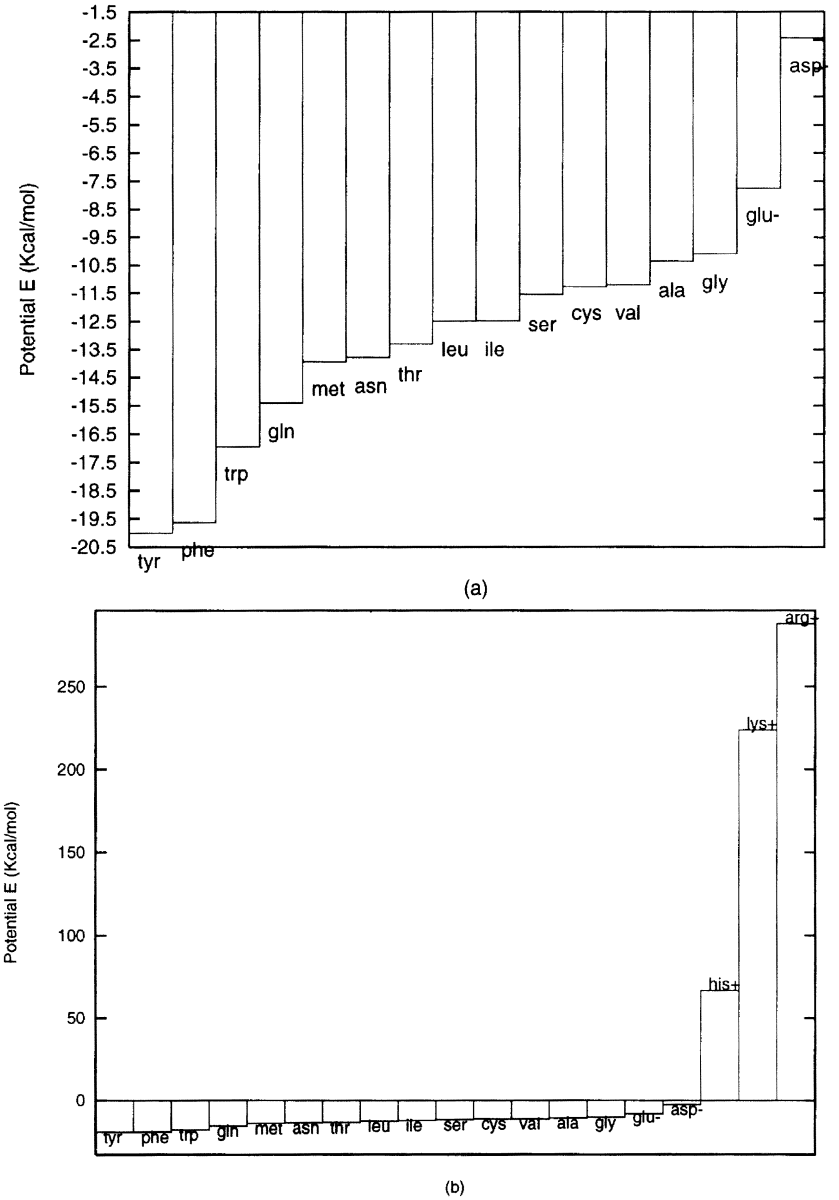


Figure 17.3. ΔE (kcal/mol) of the naturally occurring amino acids

The laboratory studies present Ser and Thr as relatively weak binders. The hydroxyl groups on both of these residues would favor interaction with polar molecules. Thus, weak interactions with the hydrophobic pocket 1 would be

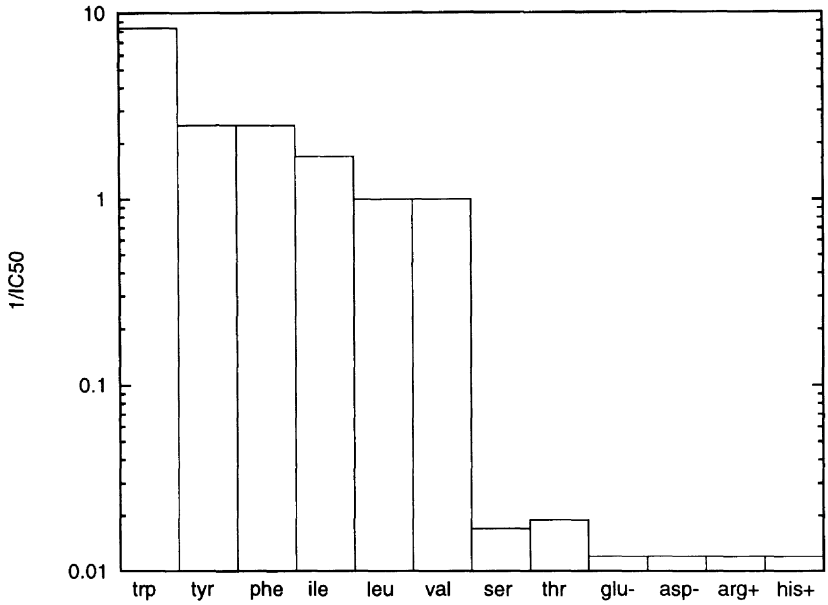


Figure 17.4. Experimental data for the naturally occurring amino acids

expected. Although, Val appears to have comparable binding energy with these two residues the optimization results support the observation that these residues are weaker binders than the small aliphatic residues. Finally, there is a number of amino acids including Gln, Lys+, Met, Asn, Cys, Gly and Ala for which no analogs were synthesized. However, it has been reported in Jardesky *et al.* (1990) and Hammer *et al.* (1994) that the peptides with Met are intermediate binders while peptides with Ala result in loss of peptide binding. The values of ΔE equal to -13.943, -10.355 for Met and Ala, respectively, found from our global optimization studies are consistent with the binding studies.

Moreover, a direct comparison with the crystallographic data can be made for the tyrosine residue that binds to pocket 1. Figure 17.5 shows the HA peptide binder (Tyrosine 308) and the global minimum conformation of tyrosine for pocket 1. An almost identical orientation is observed. Therefore, the theoretical results are in excellent agreement with those obtained by the experimental approach of competitive binding assays (Monos *et al.*, 1997).

The need for determining the global minimum conformation is illustrated in Figure 17.6 where a local minimum conformation of tyrosine corresponding to -196.637 kcal/mol is shown in comparison to the global minimum of -198.95 kcal/mol. Note that the local minimum energy conformation of tyrosine is very different than the global minimum despite its proximity to the global minimum energy value.

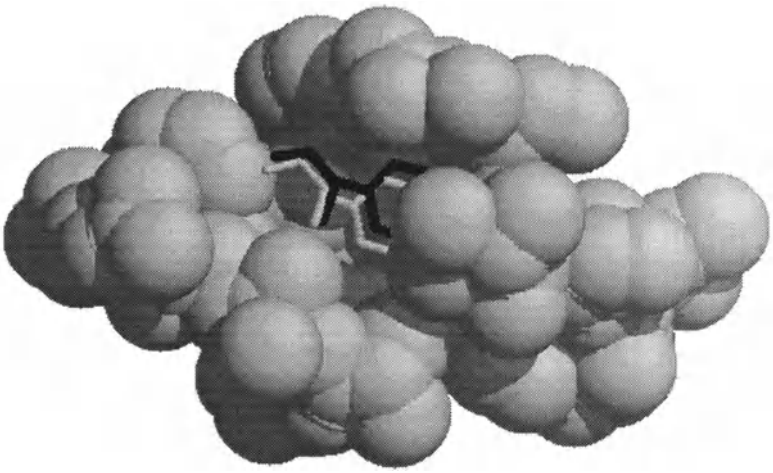


Figure 17.5. Comparison of Tyrosine binding to Pocket 1

Summary

The peptide docking problem was introduced in this chapter. After an overview of the proposed approaches for peptide docking, a novel predictive method was discussed for modeling and studying the binding affinity of different naturally occurring amino acids with pocket 1 of the HLA-DR1 protein. First, the composition of the pocket is identified together with the Cartesian coordinates of the atoms that participate in each amino acid of pocket 1 of the HLA-DR1 protein. Second, explicit relations for all the energetic *inter*-and *intra*-interactions between the atoms of the residues that define the pocket of the HLA-DR1 protein and the atoms of the considered naturally occurring amino acid were derived. Moreover, *solvation* energy was also taken into account based on a solvent accessible area method. Then, the docking problem was formulated as a nonconvex optimization problem on a set of independent dihedral angles, Euler angles and translation variables. The deterministic global optimization

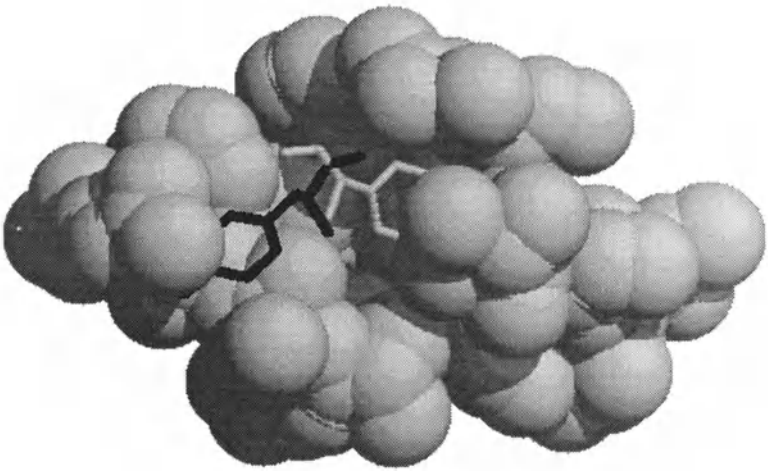


Figure 17.6. Local vs Global minimum configuration of tyrosine

method, αBB, is then used for the solution of the resulting problem. The final stage of the proposed approach consists of evaluating the interaction energy for all naturally occurring amino acids and the generation of a ranked-order list. The results of the presented approach were found to agree very well with experimental competitive binding assays.

18 THE αBB APPROACH IN BATCH DESIGN UNDER UNCERTAINTY

In this chapter, we address the batch design under uncertainty problems via the αBB global optimization framework. Section 18.1 reviews the previous research work in this area. Section 18.2 introduces the problem definition, presents the mathematical formulations, and highlights the need for global optimization. Section 18.3 presents the important theoretical properties that set the foundation for the global optimization framework, and introduces three different lower bounding formulations. Section 18.4 provides the algorithmic steps of the modified αBB global optimization approach. Section 18.5 introduces an illustrative example that identifies the characteristics of the three different lower bounding problems presented in section 18.3. Section 18.6 presents the formulation for mixed-product campaign and treats the unlimited intermediate storage (UIS) case. Section 18.7 presents computational studies for multiproduct batch plant design under uncertainty. Finally, section 18.8 provides a comparison with other types of underestimating problems. The material presented in this chapter is based on the work of Harding and Floudas (1997).

18.1 INTRODUCTION

A fundamental challenge in the design of multiproduct and multipurpose batch plants is their ability to meet production requirements and maximize profits given uncertainties both in the market demand for the products and in the operation of the process. While significant progress has been made in the de-

C. A. Floudas, *Deterministic Global Optimization*
© Springer Science+Business Media Dordrecht 2000

sign and scheduling of batch plants, until recently the issues of flexibility and design under uncertainty have received little attention. Sparrow *et al.* (1975) addressed the problem of multiproduct batch plant design via both a heuristic approach and a branch and bound formulation. This formulation assumed a single product campaign. Grossmann and Sargent (1979) presented a mixed-integer nonlinear programming (MINLP) formulation of the problem presented by Sparrow *et al.* (1975). Birewar and Grossmann (1989) incorporated the problem of scheduling in the design of multiproduct batch plants with mixed product campaigns. Voudouris and Grossmann (1992) showed that by restricting the equipment sizes to discrete values, the formulations of Grossmann and Sargent (1979) and Birewar and Grossmann (1989) could be reformulated as a series of mixed-integer linear programs (MILPs) which are more easily solved.

Among the first to address the problem of batch plant design under uncertainty in a novel way were Marketos (1975), and Johns *et al.* (1978). They divided the variables in the design problem into five categories: structural, design, state, operating, and uncertain. Structural variables describe the interconnections of the equipment in the plant. Design variables describe the size of the process equipment and are fixed once the plant is constructed. State variables are dependent variables and are determined once the design and operating variables are specified. Operating variables are those whose values can be changed in response to variations in the uncertain variables. Finally, the uncertain parameters are the quantities that can have random values which can be described by a probability distribution. Johns *et al.* (1978) also introduced the distinction between variations which have short-term effects and those with long-term effects. Wellons and Reklaitis (1989) extended this idea, suggesting a distinction between "hard" and "soft" constraints in which the former must be satisfied for feasible plant operation, but the latter may be violated, subject to a penalty in the objective function. They considered the time required to produce a product as uncertain and developed a problem formulation.

Reinhart and Rippin (1986), (1987) addressed the problem of multiproduct batch plant design with uncertainties in both demand for the products and in technical parameters such as processing times and size factors. They restricted their designs to one piece of equipment per stage. Fichtner *et al.* (1990) presented several variations on the problem of design with uncertain demands. They used interval methods to develop different solution procedures, including a two-stage approach and a penalty function approach. Shah and Pantelides (1992) proposed a scenario-based approach for the design of multipurpose batch plants with uncertain production requirements. The multipurpose approach resulted in a large-scale MILP model for which efficient techniques for obtaining good upper and lower bounds were proposed. Straub and Grossmann (1992) developed a model for the multiproduct batch design problem which takes into account uncertainties in the product demands and in equipment availability. They considered the problem of design feasibility separately from the maximization of profits and presented an approach for achieving both criteria. Subrahmanyam *et al.* (1994) addressed the problem of uncertain

demands, and used a scenario-based approach with discrete probability distributions for the demands. In addition, they considered the scheduling problem as a second stage, following the design problem. Ierapetritou and Pistikopoulos (1995), (1996) considered the multiproduct batch plant design problem based on a stochastic programming formulation. They developed a relaxation of the production feasibility requirement and added a penalty term to the objective function to account for partial feasibility. Through this analysis, the problem can be reformulated as a single large-scale nonconvex optimization problem.

18.2 MULTIPRODUCT DESIGN PROBLEM

The problem considered in this chapter is the design of a multiproduct batch plant with uncertain demands and uncertain processing parameters. The formulation of the model is similar to the models proposed in Sparrow *et al.* (1975) and Grossmann and Sargent (1979) for the production of N products in M stages in which each stage has N_j identical pieces of equipment for $j = 1, 2, \ldots, M$, with the uncertain product demands given by a known probability distribution. In this model, production occurs in a single campaign for each product and does not take into account intermediate storage.

18.2.1 Single-Product Campaign (SPC) Formulation

In this problem the objective is to maximize the profit, given by the expected revenues from the products less the investment costs of the plant equipment. The number of pieces of equipment per stage, N_j, are fixed. The equipment sizes, V_j, and the batch sizes, B_i, are the design variables, and the amounts of each product produced, Q_i, are the operating variables. $\mathcal{N}$ denotes the set of products, $\mathcal{N} = \{1, \ldots, N\}$, and $\mathcal{M}$ denotes the set of processing steps, $\mathcal{M} = \{1, \ldots, M\}$. The other parameters included in the model are, α_j and β_j, the fixed charge cost coefficients of the equipment at stage j; δ, the coefficient used to annualize the capital cost; p_i, the market price of product i; S_{ij}, the volume of the equipment in stage j needed to produce one mass unit of product i, called the size factors; t_{ij}, the amount of time needed to produce product i in one piece of equipment in stage j; T_{Li}, the amount of time in the stage which has the longest processing time for product i, called the limiting time for product i; H, the time horizon for the campaign; θ_i, the uncertain demand of product i; and V_j^L and V_j^U, the bounds on the equipment sizes available for each stage j. It should be noted that the equipment sizes, V_j, are considered as continuous variables in this model.

Ierapetritou and Pistikopoulos (1995), (1996) showed that this problem can be formulated as :

$$
\begin{aligned}
\max_{V_j, B_i} \quad & \mathbf{E}_\theta \left[\max_{Q_i} \sum_{i \in C} p_i Q_i \right] - \delta \sum_{j=1}^{M} \alpha_j N_j V_j^{\beta_j} \\
\text{subject to} \quad & V_j \geq S_{ij} B_i && \forall\, i \in \mathcal{N} \quad \forall\, j \in \mathcal{M} \\
& \sum_{i \in C} \left(\frac{Q_i}{B_i} \right) T_{Li} \leq H \\
& Q_i = \theta_i && \forall\, i \in \mathcal{N} \\
& T_{Li} = \max_{j=1,\ldots,M} \left\{ \frac{t_{ij}}{N_j} \right\} && \forall\, i \in \mathcal{N} \\
& \theta_i^L \leq \theta_i \leq \theta_i^U && \forall\, i \in \mathcal{N} \\
& V_j^L \leq V_j \leq V_j^U && \forall\, j \in \mathcal{M}.
\end{aligned}
\tag{18.1}
$$

The first set of constraints, $V_j \geq S_{ij} B_i$ are the batch size constraints. The second set of constraints, $\sum_{i \in C} \left(\frac{Q_i}{B_i} \right) T_{Li} \leq H$ are the production horizon constraints. The third set of constraints are the production feasibility constraints which require that production meet the market demand, $Q_i = \theta_i$. Finally, the fourth set of constraints, $T_{Li} = \max_{j=1,\ldots,M} \left\{ \frac{t_{ij}}{N_j} \right\}$, define the cycle time for product i as the maximum time that product i resides in any one stage.

It can be seen that the objective function contains nonconvex terms $(N_j V_j^{\beta_j})$ as do the horizon constraints $(\frac{Q_i}{B_i})$. Employing the following exponential transformations:

$$V_j = \exp(v_j) \quad \forall\, j \in \mathcal{M} \tag{18.2}$$

$$B_i = \exp(b_i) \quad \forall\, i \in \mathcal{N} \tag{18.3}$$

$$T_{Li} = \exp(t_{Li}) \quad \forall\, i \in \mathcal{N} \tag{18.4}$$

as suggested by Kocis and Grossmann (1988), convexifies the continuous part of the objective function. Next, an appropriate expression for the expected value of the revenues must be provided. The calculation of the expected revenues requires the integration over an optimization problem:

$$\mathbf{E}_\theta \left[\max_{Q_i} \sum_{i \in C} p_i Q_i \right] = \int_{\theta \in R(V_j, N_j)} \max_{Q_i} \left\{ \sum_{i \in C} p_i Q_i \right\} J(\theta) \mathrm{d}\theta \tag{18.5}$$

where $J(\theta)$ is the probability distribution function for the uncertain parameter θ. The integration should be performed over the feasible region of the plant, which is unknown at the design stage. Ierapetritou and Pistikopoulos (1995), (1996) showed that the demand constraint can be relaxed, allowing the integration to be performed in the region where the upper bound for Q_i is the value of the uncertain demand:

$$0 \leq Q_i \leq \theta_i \quad \forall\, i \in \mathcal{N}. \tag{18.6}$$

In this chapter, we consider a different relaxation of the demand constraint which involves tighter bounds on Q_i. Note that since $\theta_i^L \leq \theta_i \leq \theta_i^U$, we have:

$$\sum_{i \in C} \theta_i^L \cdot \exp(t_{Li} - b_i) \leq \sum_{i \in C} Q_i \cdot \exp(t_{Li} - b_i) \leq H. \tag{18.7}$$

In this case, the production of product i, Q_i, is allowed to vary between the lower bound for the uncertain demand for that product, θ_i^L, and the predicted demand, θ_i:

$$\theta_i^L \leq Q_i \leq \theta_i \quad \forall\, i \in \mathcal{N}. \tag{18.8}$$

When these new bounds for Q_i are used, along with the exponential transformations, (18.2), (18.3), and (18.4), the following relaxed design problem is obtained:

$$\begin{array}{lll}
\max\limits_{b_i, v_j} & \int\limits_{\theta \in R(V_j)} \max\limits_{Q_i} \left\{ \sum\limits_{i \in C} p_i Q_i \right\} J(\theta) \mathrm{d}\theta \;-\; \delta \sum\limits_{j=1}^{M} \alpha_j N_j \exp\left(\beta_j v_j\right) & \\
 & -\gamma \sum\limits_{i \in C} p_i \left(\theta_i - Q_i\right) & \\
\text{subject to} & v_j \geq \ln(S_{ij}) + b_i & \forall\, i \in \mathcal{N} \quad \forall\, j \in \mathcal{M} \\
 & t_{Li} \geq \ln\left(\frac{t_{ij}}{N_j}\right) & \forall\, i \in \mathcal{N} \quad \forall\, j \in \mathcal{M} \\
 & \sum\limits_{i \in C} Q_i \cdot \exp(t_{Li} - b_i) \leq H & \\
 & \theta_i^L \leq Q_i \leq \theta_i & \forall\, i \in \mathcal{N} \\
 & \ln(V_j^L) \leq v_j \leq \ln(V_j^U) & \forall\, j \in \mathcal{M} \\
 & \min\limits_j \ln(\frac{V_j^L}{S_{ij}^p}) \leq b_i \leq \min\limits_j \ln(\frac{V_j^U}{S_{ij}^p}) & \forall\, i \in \mathcal{N}.
\end{array} \tag{18.9}$$

Remark 18.2.1 Note that the objective function is convex with respect to the design variables, but the horizon constraints remain nonconvex. In addition, the demand constraints have been relaxed; therefore a penalty term is added to the objective function to account for the cost of unfilled orders. As the penalty parameter, γ, is increased, the production, Q_i, is driven towards the upper bound, θ_i. Due to the production horizon constraint, the batch size must be increased to accomodate the larger production. Larger batch sizes require larger equipment which increases the cost of the plant. The net result is a decrease in the expected profit.

Harding and Floudas (1997) proved the following property based upon which the bounds on the uncertain parameter values in the relaxed formulation (18.9) coincide with the feasible region of the original formulation with fixed demand, $Q_i = \theta_i$.

Property 18.2.1 *For any design, the feasibility of the batch design problem with fixed demands, θ_i, is maintained when the demand constraints are relaxed to:*

$$\theta_i^L \leq Q_i \leq \theta_i \quad \forall\, i \in \mathcal{N}. \tag{18.10}$$

Proof: See Harding and Floudas (1997).

The next step is to calculate the integral and for this purpose we apply the Gauss-Legendre quadrature formula that is discussed in the next section.

18.2.2 Gauss-Legendre Quadrature

We wish to estimate the integral:

$$\int_a^b f(x)dx. \tag{18.11}$$

We approximate the function $f(x)$ with an nth degree polynomial $p_n(x)$:

$$\int_a^b f(x)dx = \int_a^b p_n(x)dx + \int_a^b R_n(x)dx \tag{18.12}$$

where $R_n(x)$ is the error term.

Substitute for $f(x)$ using the Lagrange interpolating polynomial, $L^q(x)$, for $p_n(x)$ with its related error term:

$$f(x) = \sum_{q=0}^{n} L^q(x) f(x^q) + \prod_{q=1}^{n} (x - x^q) \frac{f^{n+1}(\xi)}{(n+1)!} \tag{18.13}$$

where,

$$L^q(x) = \prod_{j=0}^{n} \left(\frac{x - x^j}{x^q - x^j} \right) \qquad q \neq j \quad \text{and} \quad a < \xi < b. \tag{18.14}$$

To make the analysis easier, we transform the variable x,

$$v = \frac{(x-a) + (x-b)}{b-a} \tag{18.15}$$

so,

$$f(x) = F(v) = \sum_{q=0}^{n} L^q(v)F(v^q) + \left[\prod_{q=0}^{n}(v - v^q)\right] Q_n(v) \tag{18.16}$$

where $Q_n(v)$ is an nth degree polynomial and,

$$L^q(v) = \prod_{j=0}^{n} \frac{(v - v^j)}{(v^q - v^j)} \qquad q \neq j. \tag{18.17}$$

Then, the integral is given by:

$$\begin{aligned} \int_{-1}^{1} f(x)dx = \int_{-1}^{1} F(v)dv &= \int_{-1}^{1} \sum_{q=0}^{n} L^q(v)F(v^q)dv \\ &+ \int_{-1}^{1} \left[\prod_{q=0}^{n}(v - v^q)\right] Q_n(v)dv. \end{aligned} \tag{18.18}$$

Since $F(v^q)$ are fixed values, they can be taken outside the integral:

$$\begin{aligned} \int_{-1}^{1} F(v)dv &\approx \sum_{q=0}^{n} F(v^q) \int_{-1}^{1} L^q(v)dv \\ &\approx \sum_{q=0}^{n} \hat{\omega}^q \cdot F(v^q) \end{aligned} \tag{18.19}$$

where $\hat{\omega}^q$ are weighting factors and depend on the number of points n used in the approximation.

Finally, we must select the quadrature points v^q. The object is to select them in such a way so that the error term vanishes. The orthogonality property of the Legendre polynomials is used to establish the v^q values, and these can be found in tables. For the five-point quadrature formula used in this work, the weighting factors, $\hat{\omega}^q$, and the roots, v^q, are given in Table 18.1.

Note that the integration limits are -1 and 1, which is required by the Gauss-Legendre quadrature formula. However, we are interested in integrating between arbitrary numbers a and b. This can be done simply by transforming the quadrature formula to the desired interval as follows:

$$x = \frac{a(1 - v) + b(1 + v)}{2} \tag{18.20}$$

and the integral becomes,

$$\int_a^b f(x)dx = \frac{b-a}{2}\int_{-1}^{1} f\left(\frac{a(1-v)+b(1+v)}{2}\right)dv. \tag{18.21}$$

Now using the standard Gauss-Legendre quadrature formula, the right-hand integral can be approximated by,

$$\int_a^b f(x)dx \approx \frac{b-a}{2}\sum_{q=1}^{n}\hat{\omega}^q \cdot f\left(\frac{a(1-v^q)+b(1+v^q)}{2}\right). \tag{18.22}$$

For the formulations discussed in this chapter, the following substitution is made:

$$\omega^q = \frac{b-a}{2}\hat{\omega}^q \tag{18.23}$$

resulting in the following Gauss-Legendre quadrature formula:

$$\int_a^b f(x)dx \approx \frac{b-a}{2}\sum_{q=1}^{n}\omega^q \cdot f\left(\frac{a(1-v^q)+b(1+v^q)}{2}\right). \tag{18.24}$$

It still remains to calculate the joint probability density function, J, for the uncertain parameters. For a single continuous random variable, y, the normal density function, f(y) is:

$$f(y) = \frac{1}{\sigma\sqrt{2\pi}}\exp\left\{-\frac{(y-\mu)^2}{2\sigma^2}\right\} \tag{18.25}$$

where μ is the mean and σ is the standard deviation of the distribution.

In general, a multivariate normal density function is very complicated to calculate. However, since we have assumed that all uncertain parameters are independent, the joint probability density function is simply the product of the individual density functions:

$$J(y_1,\ldots,y_m) = \prod_{i=1}^{m}\frac{1}{\sigma_i\sqrt{2\pi}}\exp\left\{-\frac{(y_i-\mu_i)^2}{2\sigma_i^2}\right\} \tag{18.26}$$

and with a quadrature formulation,

$$J^q(y_1^q,\ldots,y_m^q) = \prod_{i=1}^{m}\frac{1}{\sigma_i\sqrt{2\pi}}\exp\left\{-\frac{(y_i^q-\mu_i)^2}{2\sigma_i^2}\right\}. \tag{18.27}$$

Now, for two uncertain demand parameters represented by five quadrature points each, the product $\hat{\omega}^q J^q$ is given by the 5×5 grid shown in Table 18.2. Note that the terms in the center of the grid are several orders of magnitude greater than the terms on the edges.

q	v^q	$\hat{\omega}^q$
1	-0.9061798459	0.23692885
2	-0.5384693101	0.4786286705
3	0	0.56888889
4	0.5384693101	0.4786286705
5	0.9061798459	0.23692885

Table 18.1. Five-Point Gaussian Quadrature Parameters

$\hat{\omega}^q J^q \cdot 10^3$: 5×5 Quadrature					
q_1/q_2	1	2	3	4	5
1	0.00000018	0.00002489	0.00030092	0.00002489	0.00000018
2	0.00002489	0.00352407	0.04260490	0.00352407	0.00002489
3	0.00030092	0.04260490	0.51508041	0.04260490	0.00030092
4	0.00002489	0.00352407	0.04260490	0.00352407	0.00002489
5	0.00000018	0.00002489	0.00030092	0.00002489	0.00000018

Table 18.2. Objective Function Multipliers

The uncertain processing parameters investigated in this work are the size factors, S_{ij}, and the processing times, t_{ij}. The variations in these parameters are dealt with in a scenario-based approach. In this method, the uncertain technical parameters are given different discrete values in a number of scenarios, P, while the uncertain demands retain the same probability distribution in each scenario. The scenarios are given weight factors, ω^p. The object is to find the optimal design over all possible scenarios and the optimal productions in each scenario.

Using the Gauss-Legendre quadrature approach for the uncertain demands and the scenario approach for the uncertain technical parameters results in a

nonlinear optimization problem, written as a minimization for the purposes of the following analysis:

$$
\begin{aligned}
\min_{b_i, v_j, Q_i^{qp}} \quad & \delta \sum_{j=1}^{M} \alpha_j N_j \exp(\beta_j v_j) - \sum_{p=1}^{P} \tfrac{1}{w^p} \sum_{q=1}^{Q} \omega^q J^q \sum_{i \in C} p_i Q_i^{qp} \\
& + \gamma \sum_{p=1}^{P} \tfrac{1}{w^p} \sum_{q=1}^{Q} \omega^q J^q \left\{ \sum_{i \in C} p_i \theta_i^q - \sum_{i \in C} p_i Q_i^{qp} \right\} \\
\text{subject to} \quad & v_j \geq \ln(S_{ij}^p) + b_i \qquad \forall\, i \in \mathcal{N} \;\; \forall\, j \in \mathcal{M} \;\; \forall\, p \in \mathcal{P} \\
& \sum_{i \in C} Q_i^{qp} \cdot \exp(t_{Li}^p - b_i) \leq H \quad \forall\, q \in \mathcal{Q} \quad \forall\, p \in \mathcal{P} \\
& \theta_i^L \leq Q_i^{qp} \leq \theta_i^q \qquad \forall\, i \in \mathcal{N} \quad \forall\, q \in \mathcal{Q} \quad \forall\, p \in \mathcal{P} \\
& \ln(V_j^L) \leq v_j \leq \ln(V_j^U) \qquad \forall\, j \in \mathcal{M} \\
& \min_{j,p} \ln\left(\tfrac{V_j^L}{S_{ij}^p}\right) \leq b_i \leq \min_{j,p} \ln\left(\tfrac{V_j^U}{S_{ij}^p}\right) \;\; \forall\, i \in \mathcal{N}
\end{aligned}
\tag{18.28}
$$

where

$$
t_{Li}^p = \max_j \left\{ \frac{t_{ij}^p}{N_j} \right\} \qquad \forall\, i \in \mathcal{N} \quad \forall\, p \in \mathcal{P}
$$

and each t_{Li}^p is constant because both t_{ij}^p and N_j are constant. $\mathcal{Q}$ denotes the set of quadrature points, $\mathcal{Q} = \{1, \ldots, Q\}$, and $\mathcal{P}$ denotes the set of scenarios, $\mathcal{P} = \{1, \ldots, P\}$.

In Formulation (18.28), γ is the penalty coefficient; J^q and ω^q are the probability of each quadrature point, and its weighting factor, respectively, and θ_i^q are the parameter values associated with each point. The parameter values are calculated by the formula:

$$
\theta_i^q = \frac{1}{2}\left[\theta_i^U(1 + v_i^{q_i}) + \theta_i^L(1 - v_i^{q_i})\right] \qquad \forall\, i \in \mathcal{N} \quad \forall\, q \in \mathcal{Q} \tag{18.29}
$$

where $v_i^{q_i}$ is the location of the quadrature point q_i in the $[-1, 1]$ interval, and θ_i^L, θ_i^U are the lower and upper bounds on θ_i.

Formulation (18.28) corresponds to a nonconvex NLP due to the presence of the nonconvex horizon constraints, and hence it exhibits multiplicity of solutions. This suggests that global optimization techniques must be developed to solve these problems.

Ierapetritou and Pistikopoulos (1995), (1996) applied the GOP algorithm of Floudas and Visweswaran (1990), (1993) for the case of a fixed number of equipment units per stage with relaxation constraint (18.6). In the following section, we discuss a very important theoretical property which allows very tight bounds on the global solution to be obtained in a few iterations (Harding and Floudas, 1997).

18.3 LOWER BOUNDING PROBLEM FORMULATIONS

Note that the objective function in formulation (18.28) is convex, but the horizon constraints are nonconvex. As we discussed in Chapters 11 and 12, the αBB global optimization method for finding the global solution of nonconvex problems is based on constructing a converging sequence of upper and lower bounds on the global solution. This is achieved by developing a convex relaxation of the original minimization problem by replacing all nonconvex terms with tight convex lower bounding functions. Convex lower bounding functions for twice-differentiable nonconvex terms of generic structure are constructed by means of a separable quadratic term using the α parameter developed by Maranas and Floudas (1994b). Thus, a general nonconvex term in one variable x, $NC(x)$, would become:

$$NC(x) \; + \; \alpha(x^U - x)(x^L - x) \tag{18.30}$$

where x^U, x^L are the bounds on the x variable, and α is positive. As we discussed in Chapters 11 and 12, the magnitude of the underestimator depends upon both the size of α and on the size of the current region under investigation. In Section 18.3.1, we show that for the design of multiproduct batch processes under uncertainty, an exact expression for α can be obtained. Two modifications to the α calculation method will be presented in Sections 18.3.2 and 18.3.3.

18.3.1 Formulation I : Exact α Calculation via Eigenvalue Analysis

Consider the nonconvex horizon constraints. The first observation is that the horizon constraints consist of a sum of separable terms in the form of

$$x_i \cdot \exp(y_i) \tag{18.31}$$

where,

$$x_i \; = \; Q_i^{qp} \qquad \text{and} \qquad y_i \; = \; t_{Li}^p - b_i. \tag{18.32}$$

As a result, a separate α can be calculated for each product $x_i \exp(y_i)$ in the summation. In order to find an exact expression for α, we have to obtain an exact calculation of the minimum eigenvalue. This requires the solution of the problem $|H - \lambda I| \; = \; 0$ where H is the Hessian of each separable nonconvex term and I is the identity matrix. Considering a general term:

$$\phi \; = \; x \cdot \exp(y) \tag{18.33}$$

the first derivatives are:

$$\phi_x = \exp(y) \quad \text{and} \quad \phi_y = x \cdot \exp(y) \tag{18.34}$$

and the second derivatives are:

$$\phi_{xx} = 0, \quad \phi_{xy} = \exp(y) \quad \text{and} \quad \phi_{yy} = x \cdot \exp(y). \tag{18.35}$$

Thus,

$$|H - \lambda I| = \lambda^2 - (x \exp(y))\,\lambda - (\exp(y))^2 = 0. \tag{18.36}$$

This formula yields two real solutions :

$$\lambda_1 = \left(\frac{x + \sqrt{x^2+4}}{2}\right) \exp(y) > 0 \tag{18.37}$$

$$\lambda_2 = \left(\frac{x - \sqrt{x^2+4}}{2}\right) \exp(y) < 0. \tag{18.38}$$

Therefore λ_2 is the minimum eigenvalue for all values of x and y. We need to find the minimum of λ_2 over the current domain of x and y. When x is large, the term $x - \sqrt{x^2+4}$ is negative and close to zero. The term decreases monotonically as x decreases. Therefore, λ_2 is minimized with respect to x when $x = x^L$. Since λ_2 is negative, it decreases monotonically as y increases, therefore λ_2 is minimized with respect to y when $y = y^U$. Substituting for x and y,

$$x^L = Q_i^{Lqp} \tag{18.39}$$

$$y^U = t_{Li}^p - b_i^L. \tag{18.40}$$

The resulting expression for the minimum eigenvalue is,

$$\lambda_{i,\min}^{qp} = \left(\frac{Q_i^{Lqp} - \sqrt{(Q_i^{Lqp})^2 + 4}}{2}\right) \cdot \exp(t_{Li}^p - b_i^L). \tag{18.41}$$

Since Q_i^{Lqp} is always positive λ_i^{qp} is always negative, then the corresponding α parameter is,

$$\alpha_{1,i}^{qp} = \frac{1}{4}\left(\sqrt{(Q_i^{Lqp})^2 + 4} - Q_i^{Lqp}\right) \cdot \exp(t_{Li}^p - b_i^L). \tag{18.42}$$

Remark 18.3.1 Several conclusions can be drawn from this expression. First, if Q_i^{Lqp} is an order of magnitude greater than 4, then the term within the parentheses is a small positive number. Second, if the term $(t_{Li}^p - b_i^L)$ is negative, then the exponential term is a small positive number. As will be observed from the computational studies, in the design of multiproduct batch plants under uncertainty, the aforementioned conditions are met and hence, $\alpha_{1,i}^{qp}$ is generally a very small positive number.

Using the expression for α_1, the convex lower bounding function for the production constraint is:

$$\begin{aligned} &Q_i^{qp}\exp(t_{Li}^p - b_i) + \tfrac{1}{4}\left(\sqrt{(Q_i^{Lqp})^2+4} - Q_i^{Lqp}\right)\exp\left(t_{Li}^p - b_i^L\right)\cdot \\ &\qquad \left\{(Q_i^{Lqp} - Q_i^{qp})(Q_i^{Uqp} - Q_i^{qp}) + (b_i^L - b_i)(b_i^U - b_i)\right\}. \end{aligned} \tag{18.43}$$

The resulting lower bounding problem, denoted as Formulation I, is:

$$\begin{aligned} \min_{b_i, v_j, Q_i^{qp}} \quad & \delta \sum_{j=1}^{M} \alpha_j N_j \exp(\beta_j v_j) - \sum_{p=1}^{P} \tfrac{1}{w^p} \sum_{q=1}^{Q} \omega^q J^q \sum_{i\in C} p_i Q_i^{qp} \\ & + \gamma \sum_{p=1}^{P} \tfrac{1}{w^p} \sum_{q=1}^{Q} \omega^q J^q \left\{ \sum_{i\in C} p_i \theta_i^q - \sum_{i\in C} p_i Q_i^{qp} \right\} \\ \text{s.t.} \quad & v_j \geq \ln(S_{ij}^p) + b_i \qquad \forall\, i \in \mathcal{N} \quad \forall\, j \in \mathcal{M} \quad \forall\, p \in \mathcal{P} \\ & \sum_{i\in C} Q_i^{qp} \cdot \exp(t_{Li}^p - b_i) \\ & + \sum_{i\in C} \tfrac{1}{4}\left(\sqrt{\left(Q_i^{Lqp}\right)^2 + 4} - Q_i^{Lqp}\right)\exp(t_{Li}^p - b_i^L)\cdot \\ & \left(Q_i^{Lqp} - Q_i^{qp}\right)\left(Q_i^{Uqp} - Q_i^{qp}\right) \\ & + \sum_{i\in C} \tfrac{1}{4}\left(\sqrt{\left(Q_i^{Lqp}\right)^2 + 4} - Q_i^{Lqp}\right)\exp(t_{Li}^p - b_i^L)\cdot \\ & \left(b_i^L - b_i\right)\left(b_i^U - b_i\right) \leq H \qquad \forall\, q \in \mathcal{Q} \quad \forall\, p \in \mathcal{P} \\ & \theta_i^L \leq Q_i^{qp} \leq \theta_i^q \qquad \forall\, i \in \mathcal{N} \quad \forall\, q \in \mathcal{Q} \quad \forall\, p \in \mathcal{P} \\ & \ln(V_j^L) \leq v_j \leq \ln(V_j^U) \qquad \forall\, j \in \mathcal{M} \\ & \min_{j,p} \ln\left(\tfrac{V_j^L}{S_{ij}^p}\right) \leq b_i \leq \min_{j,p} \ln\left(\tfrac{V_j^U}{S_{ij}^p}\right) \qquad \forall\, i \in \mathcal{N} \end{aligned} \tag{18.44}$$

where θ_i^q is given by Equation (18.29). The solution to Formulation I constitutes a lower bound on the solution of the original problem (18.28).

18.3.2 Formulation II : The α as a linear function

Note that in equation (18.43) the quadratic terms $(Q_i^{Uqp} - Q_i^{qp})(Q_i^{Lqp} - Q_i^{qp})$ and $(b_i^U - b_i)(b_i^L - b_i)$ are multiplied by the same α parameter. Because these terms can be of different magnitudes, it is not necessarily desirable to multiply them by the same α. To address this, a straightforward approach is to introduce a scaling factor, for example:

$$\eta_i^{qp} = \frac{Q_i^{Uqp} - Q_i^{Lqp}}{b_i^U - b_i^L} \tag{18.45}$$

and define a different α for each variable. Thus, the lower bounding function for a generic nonconvex term of two variables would be:

$$NC(x,y) + \alpha(x^U - x)(x^L - x) + \beta(y^U - y)(y^L - y) \tag{18.46}$$

where

$$\beta = \eta \cdot \alpha. \tag{18.47}$$

For the production constraints, which have the form $NC(x,y) = x\exp(y)$, the lower bounding function, L, can be written:

$$L = x\exp(y) + \frac{\alpha}{\eta}(x^U - x)(x^L - x) + \alpha(y^U - y)(y^L - y). \tag{18.48}$$

First, consider η to be an arbitrary positive constant and attempt to find an expression for a constant α that maintains the convexity of L. The lower bounding function is convex if both of the eigenvalues of the Hessian of L are always positive. Both eigenvalues are positive if the following two conditions are met:

(i) $\frac{\partial^2 L}{\partial x^2} + \frac{\partial^2 L}{\partial y^2} \geq 0$

(ii) $\frac{\partial^2 L}{\partial x^2}\frac{\partial^2 L}{\partial y^2} - (\frac{\partial^2 L}{\partial x \partial y})^2 \geq 0$

We find that α of the form,

$$\hat{\alpha} = \frac{1}{4}\left(\sqrt{(x^L)^2 + 4\eta} - x^L\right)\exp(y^U) \tag{18.49}$$

guarantees that L is convex for any $\eta > 0$.

Harding and Floudas (1997) proved that the scaling factor can be written as:

$$\eta_i^{qp} = \max\left\{1, \frac{(Q_i^{Uqp} - Q_i^{Lqp})^2}{(b_i^U - b_i^L)^2} - Q_i^{Lqp}\frac{(Q_i^{Uqp} - Q_i^{Lqp})}{(b_i^U - b_i^L)}\right\} \tag{18.50}$$

and hence the α term is:

$$\hat{\alpha}_i^{qp} = \frac{1}{4}\left(\sqrt{(Q_i^{Lqp})^2 + 4\eta_i^{qp}} - Q_i^{Lqp}\right)\exp(t_{Li}^p - b_i^L). \tag{18.51}$$

Recall that $\hat{\alpha}$ is a constant, and thus is constrained by the most restrictive point, (x^L, y^U), while at other points in the feasible region it may be sufficient to use a smaller value of α. The goal is to find an expression for α that is a function of the variables that maintains the convexity of the lower bounding function.

Consider the simplest function of x and y, that is, a linear function. Now the lower bounding function has the form:

$$L = x\exp(y) + \frac{\alpha_2(x)}{\eta}\cdot(x^U - x)(x^L - x) + \beta_2(y)\cdot(y^U - y)(y^L - y) \tag{18.52}$$

where,

$$\begin{aligned} \alpha_2(x) &= \hat{\alpha}\left(1 - c\cdot\tfrac{x - x^L}{x^U - x^L}\right) \\ \beta_2(y) &= \hat{\alpha}\left(1 - d\cdot\tfrac{y^U - y}{y^U - y^L}\right) \\ 0 &\le c, d \le 1 \end{aligned} \tag{18.53}$$

where c and d are constants. Note that at the point where the minimum eigenvalue as defined by Equation (18.41) is minimized, (x^L, y^U), the expressions for α_2 and β_2 reduce to:

$$\alpha_2(x^L) = \hat{\alpha} \tag{18.54}$$

$$\beta_2(y^U) = \hat{\alpha} \tag{18.55}$$

and at any other point, say (x^U, y^L),

$$\alpha_2(x^U) = \hat{\alpha}(1 - c) \le \hat{\alpha} \tag{18.56}$$

$$\beta_2(y^L) = \hat{\alpha}(1 - d) \le \hat{\alpha} \tag{18.57}$$

The constants, c and d, must be greater than zero or else no improvement in the underestimator is made. In addition, the constants must be less than or equal to unity or else the term is not a valid underestimator. The goal is to

find the largest values of c and d which maintain the convexity of the lower bounding function. A procedure for determining the upper bounds on c and d is as follows:

(1) Let $c = d = \frac{1}{2}\left[1 - \frac{\exp(y^L)}{4\hat{\alpha}}\left(\sqrt{(x^U)^2 + 4\eta} - x^U\right)\right]$.

(2) Check $c \leq \frac{1}{2}\left[1 - \frac{\eta \exp(2y^U)}{4\hat{\alpha}^2\left(\frac{x^U \exp(y^U)}{2\hat{\alpha}} + 1 + d\right)}\right]$.

Check $d \leq \frac{1}{2}\left(\frac{x^L \exp(y^L)}{2\hat{\alpha}} + 1 - \frac{\eta \exp(2y^L)}{4\hat{\alpha}^2(1+c)}\right)$.

If both are satisfied, then Stop.
If one or more is not satisfied, then go to Step 3.

(3) $c = \min\left\{\frac{1}{2}\left[1 - \frac{\eta \exp(2y^U)}{4\hat{\alpha}^2\left(\frac{x^U \exp(y^U)}{2\hat{\alpha}} + 1 + d\right)}\right], \frac{1}{2}\left[1 - \frac{\eta \exp(2y^L)}{4\hat{\alpha}^2\left(\frac{x^U \exp(y^L)}{2\hat{\alpha}} + 1 - 2d\right)}\right]\right\}$.

(4) $d = \min\left\{1, \frac{1}{2} + \frac{x^L \exp(y^L)}{4\hat{\alpha}} - \frac{\eta \exp(2y^L)}{8\hat{\alpha}^2(1+c)}, \frac{1}{2} + \frac{x^U \exp(y^L)}{4\hat{\alpha}} - \frac{\eta \exp(2y^L)}{8\hat{\alpha}^2(1-2c)}\right\}$.

(5) Return to Step 3 and repeat until c and d converge.

A derivation of this procedure can be found in Harding and Floudas (1997). When the Q and b variables are substituted for x and y, the α parameters are written:

$$\begin{aligned} \alpha_{2,i}^{qp}(Q_i^{qp}) &= \hat{\alpha}_i^{qp}\left(1 - c_i^{qp} \cdot \frac{Q_i^{qp} - Q_i^{L\,qp}}{Q_i^{U\,qp} - Q_i^{L\,qp}}\right) \\ \beta_{2,i}^{qp}(b_i) &= \hat{\alpha}_i^{qp}\left(1 - d_i^{qp} \cdot \frac{b_i - b_i^L}{b_i^U - b_i^L}\right). \end{aligned} \tag{18.58}$$

Using the new expression for the lower bounding terms with α_2 and β_2, Formulation II of the lower bounding problem is:

$$
\begin{aligned}
\min_{b_i, v_j, Q_i^{qp}} \quad & \delta \sum_{j=1}^{M} \alpha_j N_j \exp(\beta_j v_j) - \sum_{p=1}^{P} \frac{1}{w^p} \sum_{q=1}^{Q} \omega^q J^q \sum_{i \in C} p_i Q_i^{qp} \\
& + \gamma \sum_{p=1}^{P} \frac{1}{w^p} \sum_{q=1}^{Q} \omega^q J^q \left\{ \sum_{i \in C} p_i \theta_i^q - \sum_{i \in C} p_i Q_i^{qp} \right\} \\
\text{subject to} \quad & v_j \geq \ln(S_{ij}^p) + b_i \qquad \forall\, i \in N \quad \forall\, j \in M \quad \forall\, p \in P \\
& \sum_{i \in C} \Bigg\{ Q_i^{qp} \cdot \exp(t_{Li}^p - b_i) \\
& \quad + \frac{1}{4\eta_i^{qp}} \left(\sqrt{(Q_i^{Lqp})^2 + 4\eta_i^{qp}} - Q_i^{Lqp} \right) \exp(t_{Li}^p - b_i^L) \\
& \quad \cdot \left(1 - c_i^{qp} \cdot \frac{Q_i^{qp} - Q_i^{Lqp}}{Q_i^{Uqp} - Q_i^{Lqp}} \right) \cdot \left(Q_i^{qp,L} - Q_i^{qp} \right) \left(Q_i^{qp,U} - Q_i^{qp} \right) \\
& \quad + \frac{1}{4} \left(\sqrt{(Q_i^{Lqp})^2 + 4\eta_i^{qp}} - Q_i^{Lqp} \right) \exp(t_{Li}^p - b_i^L) \\
& \quad \cdot \left(1 - d_i^{qp} \cdot \frac{b_i - b_i^L}{b_i^U - b_i^L} \right) \left(b_i^L - b_i \right) \left(b_i^U - b_i \right) \Bigg\} \leq H \qquad \forall\, q \quad \forall\, p \\
& \theta_i^L \leq Q_i^{qp} \leq \theta_i^q \qquad \forall\, i \in N \quad \forall\, q \in Q \quad \forall\, p \in P \\
& \ln(V_j^L) \leq v_j \leq \ln(V_j^U) \qquad \forall\, j \in M \\
& \min_j \ln\left(\frac{V_j^L}{S_{ij}^p}\right) \leq b_i \leq \min_j \ln\left(\frac{V_j^U}{S_{ij}^p}\right) \qquad i = 1, \ldots, N.
\end{aligned}
\tag{18.59}
$$

18.3.3 *Formulation III : Nonconvex Lower Bounding Function*

In the derivation of α_1, the minimum eigenvalue of the Hessian matrix of the nonconvex term $Q_i^{qp} \cdot \exp(t_{Li}^p - b_i)$ was found to be:

$$
\lambda_i^{qp} = \left(\frac{Q_i^{qp} - \sqrt{(Q_i^{qp})^2 + 4}}{2} \right) \cdot \exp(t_{Li}^p - b_i). \tag{18.60}
$$

The minimum of this expression over the current search domain is used to calculate the constant value for α_1. We have noted some interesting results if Equation (18.60) rather than its minimum is used to calculate the α parameter. In this case, the expression for α is:

$$
\alpha_{3,i}^{qp} = \frac{1}{4} \left(\sqrt{(Q_i^{qp})^2 + 4} - Q_i^{qp} \right) \cdot \exp(t_{Li}^p - b_i). \tag{18.61}
$$

Note that α_3 does not strictly satisfy the convexity condition for the whole domain. Therefore, the lower bounding function constructed using α_3 is not necessarily convex. Thus, the lower bounding problem is still a nonconvex problem and local solvers may converge to a local minimum rather than the global minimum, giving an overestimate of the lower bound. In practice, however, the algorithm never failed to converge to the correct global minimum solution when α_3 was used to construct the lower bounding function.

18.4 GLOBAL OPTIMIZATION ALGORITHM - MODIFIED αBB

A modified version of the αBB algorithm presented in Chapters 11 and 12 is used to solve the multiproduct batch plant design problems. The goal of this procedure is to locate the global minimum solution by constructing a sequence of converging upper and lower bounds. The approach will be described for the design problem that employs the convex underestimator (18.44).

18.4.1 Algorithmic Description

The basic steps of the algorithm are the following (only the formulation for the $\boldsymbol{b}$ variables is shown):

STEP 0 - Initialization

The relative convergence tolerance, ϵ, is specified and the iteration counter, *Iter*, is set to one.

For the first iteration,

STEP 1 - Variable Bound Refinement

The global bounds on the $\boldsymbol{v}$ and $\boldsymbol{b}$ variables are tightened by solving the following problems (only the formulation for the $\boldsymbol{b}$ variables is shown):

$$
\begin{aligned}
b_i^L \;=\; \min \quad & b_i \\
\text{s.t.} \quad & v_j \;\geq\; \ln(S_{ij}^p) \;+\; b_i \qquad \forall\, i \in N \quad \forall\, j \in M \quad \forall\, p \in P \\
& \sum_{i \in C} \Big\{ Q_i^{qp} \cdot \exp(t_{Li}^p - b_i) \\
& \quad + \alpha_{1,i}^{qp} \cdot \left(Q_i^{qp,L} - Q_i^{qp}\right)\left(Q_i^{qp,U} - Q_i^{qp}\right) \\
& \quad + \alpha_{1,i}^{qp} \cdot \left(b_i^L - b_i\right)\left(b_i^U - b_i\right) \Big\} \;\leq\; H \qquad \forall\, q \quad \forall\, p \\
& \theta_i^L \;\leq\; Q_i^{qp} \;\leq\; \theta_i^q \qquad \forall\, i \in N \quad \forall\, q \in Q \quad \forall\, p \in P \\
& \ln(V_j^L) \;\leq\; v_j \;\leq\; \ln(V_j^U) \qquad \forall\, j \in M \\
& \min_j \ln\left(\tfrac{V_j^L}{S_{ij}^p}\right) \;\leq\; b_i \;\leq\; \min_j \ln\left(\tfrac{V_j^U}{S_{ij}^p}\right) \qquad \forall\, i \in N
\end{aligned}
\tag{18.62}
$$

and,

$$
\begin{aligned}
b_i^U \;=\; \max \quad & b_i \\
\text{s.t.} \quad & v_j \;\geq\; \ln(S_{ij}^p) \;+\; b_i \qquad \forall\, i \in N \quad \forall\, j \in M \quad \forall\, p \in P \\
& \sum_{i \in C} \Big\{ Q_i^{qp} \cdot \exp(t_{Li}^p - b_i) \\
& \quad + \alpha_{1,i}^{qp} \cdot \left(Q_i^{qp,L} - Q_i^{qp}\right)\left(Q_i^{qp,U} - Q_i^{qp}\right) \\
& \quad + \alpha_{1,i}^{qp} \cdot \left(b_i^L - b_i\right)\left(b_i^U - b_i\right) \Big\} \;\leq\; H \qquad \forall\, q \quad \forall\, p \\
& \theta_i^L \;\leq\; Q_i^{qp} \;\leq\; \theta_i^q \qquad \forall\, i \in N \quad \forall\, q \in Q \quad \forall\, p \in P \\
& \ln(V_j^L) \;\leq\; v_j \;\leq\; \ln(V_j^U) \qquad \forall\, j \in M \\
& \min_j \ln\left(\tfrac{V_j^L}{S_{ij}^p}\right) \;\leq\; b_i \;\leq\; \min_j \ln\left(\tfrac{V_j^U}{S_{ij}^p}\right) \qquad \forall\, i \in N
\end{aligned}
\tag{18.63}
$$

The current variable bounds $\left[\boldsymbol{b}^{L,Iter}, \boldsymbol{b}^{U,Iter}, \boldsymbol{v}^{L,Iter}, \boldsymbol{v}^{U,Iter}, \boldsymbol{Q}^{L,Iter}, \boldsymbol{Q}^{U,Iter}\right]$, are then set equal to the global bounds $\left[\boldsymbol{b}^L, \boldsymbol{b}^U, \boldsymbol{v}^L, \boldsymbol{v}^U, \boldsymbol{Q}^L, \boldsymbol{Q}^U\right]$.

STEP 2 - Upper Bound on Global Solution

An initial upper bound (UB) on the global solution is obtained by solving the overall problem locally. The variable values at the solution are stored: $\boldsymbol{b}^{UB} = \boldsymbol{b}^*, \boldsymbol{v}^{UB} = \boldsymbol{v}^*, \boldsymbol{Q}^{UB} = \boldsymbol{Q}^*$.

STEP 3 - Lower Bound on Global Solution

An initial lower bound (LB) on the global solution is obtained by solving the convexified problem (18.44) within the refined variable bounds. The variable values at the solution are stored: $\boldsymbol{b}^{*,Iter}, \boldsymbol{v}^{*,Iter}, \boldsymbol{Q}^{*,Iter}$.

For all subsequent iterations (Iter $\geq$ 2),

STEP 4 - Check for Convergence

If the relative difference between the upper and lower bound is greater than the convergence tolerance, $(UB - LB) \,/\, |UB + 1| \;>\; \epsilon$, then continue to STEP 5. Otherwise, the global solution has been found and the solution is:

$$
\begin{aligned}
f^* \;&\leftarrow\; f^{UB} \\
\boldsymbol{b}^* \;&\leftarrow\; \boldsymbol{b}^{UB} \\
\boldsymbol{v}^* \;&\leftarrow\; \boldsymbol{v}^{UB} \\
\boldsymbol{Q}^* \;&\leftarrow\; \boldsymbol{Q}^{UB}
\end{aligned}
\tag{18.64}
$$

STEP 5 - Update Current Region and Lower Bound

The current search region is selected as the region containing the minimum of all stored lower bound solutions. The lower bound, LB, is updated to be the minimum of all stored solutions. The current variable bounds are updated to

be the bounds of the current region, and the current point is updated to be the solution point of the current region.

STEP 6 - Update Upper Bound on Global Solution

The current point is used as a starting point for a local solution of the problem. If the solution to this problem is less than the current upper bound, then the upper bound is updated to the new solution, and the variable values at the new upper bound are stored.

STEP 7 - Select Branching Variable and Partition Current Region

The current region is partitioned into two subregions by bisecting the edge corresponding to a chosen variable. The criterion for choosing the branching variable is discussed in Section 18.4.2.

STEP 8 - Solve Lower Bounding Problems

The lower bounding problem (18.44) is solved in both new subregions. If the solution in a region is greater than the current upper bound, then this region is guaranteed not to contain the global minimum and it is fathomed. Otherwise, the solution is stored along with the variable bounds for the region and the variable values at the solution. Return to STEP 4.

18.4.2 Analysis of the Branching Criteria

In the implementation of the αBB algorithm for the example problems, it was found that the speed of convergence was greatly affected by the choice of variables on which the subregions were partitioned (branching variable). In this section, studies on different branching criteria are presented. First, it was observed that the $\boldsymbol{v}$ variables do not participate in the objective function or in any nonlinear terms and hence they should not be considered in the choice of branching variables.

The first branching criterion examined was the most straightforward one. In the current region, the variable with the longest distance between its upper and lower bounds $(x^U - x^L)$, or the longest relative distance $(2(x^U - x^L)/(x^U + x^L))$ is chosen as the branching variable. This choice of criterion follows the reasoning that since the size of the approximation, $\alpha(x^L - x)(x^U - x)$, depends on the distance between x^L and x^U, then by partitioning on the variable with the largest range, we can hopefully achieve the largest improvement in the approximation. In practice, it was found that this procedure converges to the global solution slowly.

In the second branching criterion, the entire convex lower bounding function was calculated for each variable at the solution point in the current region.

$$\alpha_{1,i}^{qp} \cdot \left(Q_i^{qp,L} - Q_i^{qp}\right)\left(Q_i^{qp,U} - Q_i^{qp}\right) \tag{18.65}$$

or

$$\alpha_{1,i}^{qp} \cdot \left(b_i^L - b_i\right)\left(b_i^U - b_i\right). \tag{18.66}$$

The variable with the smallest value of the underestimation term (all terms are ≤ 0) is chosen as the branching variable.

Several observations were made when this criterion was used for the solution of the Illustrative Example discussed in section 18.5. First, this approach also converged to the global solution, but only slightly faster than the first method. Second, branching on the $\boldsymbol{Q}$ variables caused the largest improvement in the lower bound. Third, it was found that only $\boldsymbol{Q}$ variables were chosen as branching variables until the last iteration before convergence, and then a $\boldsymbol{b}$ variable was chosen.

Based on these observations, the algorithm was implemented where only branching on the $\boldsymbol{Q}$ variables was allowed, and then branching in the $\boldsymbol{b}$ variables was allowed. It was found that the algorithm did not converge when only the $\boldsymbol{b}$ variables were used as the branching variables. Further, it was observed that the algorithm converged in the same number of iterations when only the $\boldsymbol{Q}$ variables were used as when both the $\boldsymbol{b}$ and $\boldsymbol{Q}$ variables were used.

A third possible branching criterion is to branch only on the variables that have the largest multipliers in the objective function. The values of the probability distribution, J^q, and the weighting factors, ω^q, for the quadrature points vary over several orders of magnitude. As a result, some of the $\boldsymbol{Q}$ variables have a much larger effect on the objective function than others. Table 18.2 shows the product $\omega^q J^q$ for a 5×5 quadrature grid for a plant that produces two products, like the Illustrative Example and Example 1. As can be seen, the multiplier corresponding to the $Q_i^{3,3,p}$ variables is the largest, followed by $Q_i^{3,2,p}, Q_i^{2,3,p}, Q_i^{3,4,p}, and Q_i^{4,3,p}$.

Initially, only the $Q_i^{3,3,p}$ variables were used as the branching variables, using the underestimation criterion, Equation (18.65), to select the variable to branch on. An implementation where the $Q_i^{3,3,p}, Q_i^{3,2,p}, Q_i^{2,3,p}, Q_i^{3,4,p}$, and $Q_i^{4,3,p}$ variables were selected as possible branching variables was also used. However, it was found that neither of these implementations converged because the convex lower bounding functions corresponding to these variables were reduced to zero before the desired tolerance was achieved.

Finally, a combination of the second and third strategies was used as the branching criterion. The product of the objective function multiplier, $\omega^q J^q$ and the underestimation, Equation (18.65), is calculated for all $\boldsymbol{Q}$ variables. This directly determines the effect of the convex lower bounding function on the objective function. Thus, the variable with the largest product is chosen as the branching variable. This procedure was found to converge in fewer iterations than all other methods discussed above. Note that this corresponds to branching on all $\boldsymbol{Q}$ variables first and then branching on the $\boldsymbol{b}$ variables next, if needed.

18.5 ILLUSTRATIVE EXAMPLE

In order to illustrate that a tight lower bound of the global solution can be obtained immediately by using the exact expression for α_1, consider a small example taken from Grossmann and Sargent (1979) for the design of a batch plant that must produce two products using three stages with one piece of equipment per stage. In this example, only the demands are considered to be uncertain parameters, therefore there is only one scenario ($P = 1$). The data for this example are shown in Table 18.3.

The single-product campaign (SPC) formulation, (18.44), contains fifty-five variables and thirty-one constraint equations, not counting the variable bounds. Of the thirty-one constraints, twenty-five are nonlinear constraints containing fifty nonlinear terms. The demands have a normal distribution of N(200,10) and N(100,10) for products 1 and 2, respectively. Five quadrature points are used for the uncertain demand of each product. The upper and lower bound on the equipment volumes, V_j, were 4500 and 500, respectively. A relative tolerance, ϵ, of 0.003 was used in this example, and the time horizon, H, was 8. The annual cost coefficient, δ, was 0.3.

The optimal design and profit is shown in Table 18.4, and the progression of the upper and lower bounds on the global solution for the case where the penalty term is zero is shown in Table 18.5. This method provides a very tight lower bound, as the lower bound is within one percent of the global solution at the first iteration, without any partitioning of the search domain. The branch and bound algorithm quickly converges to the desired tolerance.

Note that the equipment and batch sizes for the global solution increase as the γ coefficient increases. This reflects the penalty associated with not fully meeting the market demand. In addition, the computational requirements and iterations decrease as the penalty coefficient increases.

Table 18.6 shows the results for the Illustrative Example using Formulation II and Table 18.7 shows the progression of the upper and lower bounds on the global solution.

Note in Table 18.7 that the initial lower bound is -986.777. For the formulation using α_1, the initial lower bound from Table 18.5 was -987.840. Thus α_2 provides a tighter lower bound on the global solution than α_1. This point is further illustrated in Table 18.6 where the number of iterations required to achieve the desired tolerance is less for α_2 than for the lower bounding functions using α_1 (Table 18.4) when $\gamma = 0$ and $\gamma = 8$. However, the CPU time required is slightly greater, most likely due to the more complicated expression for the convex lower bounding function when α_2 is used.

The optimal solutions shown in Table 18.6 are exactly the same for the α_1 formulation shown in Table 18.4, as expected.

The advantage of using α_3 is that it provides a much tighter lower bound on the global solution than either α_1 or α_2. These observations are demonstrated below in the Illustrative Example and in the other examples in Section 24.5. The number of iterations and the CPU time required to converge to the global solution are shown in Table 18.8. Fewer iterations were required to achieve

the desired tolerance when α_3 was used than when α_1 and α_2 were used. In addition, the computational requirements were less when α_1 was used in the lower bounding function.

Table 18.9 shows the progressive improvement of the bounds using the proposed approach. As can be seen, the initial lower bound is very close to the global solution and the algorithm converges quickly. Recall from Table 18.5 that the initial lower bound when α_1 is used was -987.840. An improved initial lower bound of -986.777 is provided by α_2, Table 18.7. A much larger improvement is possible when α_3 is used, resulting in an initial lower bound of -982.171.

Size Factors				Processing Times			Cost Coefficients			Prices of
	Stage				Stage		Stage	α_j	β_j	Products
Product	1	2	3		1	2	3	1	5	0.6	p_i
1	2	3	4		8	20	8	2	5	0.6	5.5
2	4	6	3		16	4	4	3	5	0.6	7.0

Table 18.3. Data for Illustrative Example

γ	Optimal Profit	Optimal Design: α_1					Iterations	CPU s
		V_1	V_2	V_3	B_1	B_2		
Single-Product Campaign								
0	979.186	1800	2700	3600	900	450	4	1.30
4	937.424	1908	2861	3815	954	477	4	0.86
8	934.854	1972	2958	3945	986	493	3	0.59

Table 18.4. Results for Illustrative Example (SPC)

Remark 18.5.1 In many cases it is possible to eliminate unnecessary steps in the production of one or more products, or to use some equipment for different processing tasks. In this case, the plant is called a multipurpose batch plant. The formulation of the multipurpose batch plant design problem under uncertainty requires few modifications of the multiproduce batch design problem, and does not introduce any new nonconvex terms. Harding and Floudas (1997) showed that the analysis for the lower bounding terms presented in Section 18.3 can be directly applied to the nonconvex terms in the multipurpose batch design problem.

α_1			
Iteration	Upper Bound	Lower Bound	Relative Difference
1	-979.180	-987.840	0.00884
2	-979.185	-984.106	0.00502
3	-979.186	-982.404	0.00328
4	-979.186	-981.278	0.00213

Table 18.5. Progression of Upper and Lower Bounds for Illustrative Example, ($\gamma = 0$)

γ value	Optimal Profit	Optimal Design: α_2					Iters	CPU s
		V_1	V_2	V_3	B_1	B_2		
Single-Product Campaign								
0	979.186	1800	2700	3600	900	450	3	1.47
4	937.424	1908	2861	3815	954	477	4	1.90
8	934.854	1972	2958	3945	986	493	2	0.71

Table 18.6. Results for Illustrative Example (SPC) for α_2

$\alpha_2(Q), \beta_2(b)$			
Iteration	Upper Bound	Lower Bound	Relative Difference
1	-979.180	-986.777	0.00775
2	-979.185	-983.599	0.00451
3	-979.186	-982.093	0.00297

Table 18.7. Progression of Upper and Lower Bounds for Illustrative Example, ($\gamma = 0$)

18.6 MIXED-PRODUCT CAMPAIGN FORMULATION

As stated in Section 18.2, Formulation (18.28) corresponds to the multiproduct batch design problem with single-product campaigns and no intermediate

γ	Optim. Profit	Optimal Design: α_3					Iters	CPU s
		V_1	V_2	V_3	B_1	B_2		
Single-Product Campaign								
0	979.186	1800	2700	3600	900	450	2	0.75
4	937.424	1908	2861	3815	954	477	2	0.56
8	934.854	1972	2958	3945	986	493	1	0.18

Table 18.8. Results for Illustrative Example (SPC)

$\alpha_3(Q,b)$			
Iteration	Upper Bound	Lower Bound	Relative Difference
1	-979.180	-982.171	0.00305
2	-979.186	-980.880	0.00173

Table 18.9. Progression of Upper and Lower Bounds for Illustrative Example, ($\gamma = 0$)

storage. Since this formulation uses the maximum processing time for each product, $t_{Li} = \max_{j=1,\dots,M} \left\{ \frac{t_{ij}}{N_j} \right\}$, it overestimates the time required to process each product. This can result in an overdesign of the equipment sizes.

In order to allow mixed-product campaigns, the horizon constraint in Formulation (18.28) must be replaced by an expression that takes into account the processing times in each stage, rather than the maximum processing time. This can be done in the following manner according to the analysis of Birewar and Grossmann (1989). Let n_i be the number of batches of product i over the whole production period:

$$n_i = \frac{Q_i}{B_i}. \tag{18.67}$$

The total cycle time for each stage, CT_j^{tot}, must be at least as large as the sum over all products of the processing time for each product i in stage j.

$$CT_j^{tot} \geq \sum_{i \in C} n_i t_{ij} \qquad j = 1, \dots, M. \tag{18.68}$$

Obviously, the total cycle time in each stage must be less than the horizon time, so:

$$H \;\geq\; CT_j^{tot} \;\geq\; \sum_{i \in C} n_i t_{ij} \qquad j \;=\; 1, \ldots, M. \tag{18.69}$$

Substituting for n_i results in the following expression for the new horizon constraint:

$$\sum_{i \in C} \frac{Q_i}{B_i} t_{ij} \;\leq\; H \qquad j \;=\; 1, \ldots, M. \tag{18.70}$$

Applying the exponential transformation, the Gaussian quadrature, and the scenario analysis presented in Section 18.2 gives the final expression for the new constraint:

$$\sum_{i \in C} Q_i^{qp} \cdot \exp\left\{t_{ij}^p \;-\; b_i\right\} \;\leq\; H. \tag{18.71}$$

Therefore, the global optimization problem for the design of multiproduct batch plants under uncertainty with unlimited intermediate storage (UIS) and zero cleanup times is the following:

$$\begin{aligned}
\min_{v_j, Q_i^{qp}} \quad & \delta \sum_{j=1}^{M} \alpha_j N_j \exp\left(\beta_j v_j\right) \\
& - \sum_{p=1}^{P} \frac{1}{w^p} \sum_{q=1}^{Q} \omega^q J^q \sum_{i \in C} p_i Q_i^{qp} \\
& + \gamma \sum_{p=1}^{P} \frac{1}{w^p} \sum_{q=1}^{Q} \omega^q J^q \left\{ \sum_{i \in C} p_i \theta_i^q \;-\; \sum_{i \in C} p_i Q_i^{qp} \right\} \\
\text{subject to} \quad & v_j \;\geq\; \ln(S_{ij}^p) \;+\; b_i \qquad \forall\, i \in N \;\; \forall\, j \in M \;\; \forall\, p \in P \\
& \sum_{i \in C} Q_i^{qp} \cdot \exp(t_{ij}^p - b_i) \;\leq\; H \;\; \forall\, j \in M \;\; \forall\, q \in Q \;\; \forall\, p \in P \\
& \theta_i^L \;\leq\; Q_i^{qp} \;\leq\; \theta_i^q \qquad \forall\, i \in N \;\; \forall\, q \in Q \;\; \forall\, p \in P \\
& \ln(V_j^L) \;\leq\; v_j \;\leq\; \ln(V_j^U) \quad \forall\, j \in M.
\end{aligned} \tag{18.72}$$

Note that the cost of storage facilities is neglected and the batch sizes for each product are assumed to be the same at each stage. In addition, there are now $M \cdot Q \cdot P$ horizon constraints as opposed to $Q \cdot P$ horizon constraints for Formulation (18.28). This greatly increases the size of the problem.

For the convex lower bounding problem, the horizon constraints are replaced by their convex underestimators:

$$\sum_{i \in C} Q_i^{qp} \exp(t_{ij} - b_i) + \alpha_{2,i}^{qp}(Q_i^{qp,L} - Q_i^{qp})(Q_i^{qp,U} - Q_i^{qp}) + \beta_{2,i}^{qp}(b_i^L - b_i)(b_i^U - b_i).$$

Since each t_{ij} is used rather than the maximum over all stages, t_{Li}, the size of the exponential term in the underestimator will be larger in general than for the single-product campaign model. Due to the increased size of the UIS problem it is expected that the number of iterations needed to solve this problem and the CPU time per iteration will increase.

18.7 COMPUTATIONAL STUDIES

The following example problems illustrate the advantage of using the tight convex lower bounding functions developed in this work. The global solutions were found in a small number of iterations, even for very large problems. The algorithm presented in this section was implemented in GAMS (Brooke *et al.*, 1992) and MINOS5.4 (Murtagh and Saunders, 1983) was used as a local solver. An IBM RS6000 was used to run all examples, and CPU times are reported in seconds. For all examples, five quadrature points for each product were used in the Gaussian Quadrature approximation of the expected profit integral.

Example 1. This example is similar to the Illustrative Example, but in this case, the size factors and processing times are taken to be uncertain parameters. Three scenarios are used to provide a range of values for the uncertain parameters. The additional scenarios increase the size of the problem. In the SPC problem formulation there are 155 variables and 93 constraint equations. These include 75 nonlinear constraints and 150 nonlinear terms. The UIS formulation also has 155 variables, but contains 225 nonlinear constraints with 450 nonlinear terms, and a total of 243 constraints. The weighting factor, w^p, was the same for each scenario and had a value of $\frac{1}{3}$. A relative tolerance of 0.003 was used in this example and the horizon time, H, was 8. The data are given in Tables 18.10 and 18.11, and the results are shown in Tables 18.12 and 18.13.

Table 18.13 shows that the UIS scheduling solution results in a substantial decrease in equipment and batch sizes. The decrease in equipment sizes causes an increase in the profit.

Comparing the results of Example 1 to the Illustrative Example, an increase in the CPU time is seen, which is due to the larger size of the problem in Example 1. The CPU time required for each iteration is higher for the UIS problem than for the SPC problem, but it is interesting that the number of iterations required for the UIS problem is one fewer than for the SPC problem using α_2 with $\gamma = 0$. Note that using α_2 gives a tighter initial lower bound than α_1, and requires fewer iterations to converge to the global solution. Further, as was shown in Section 18.3.3 with the Illustrative Example, using α_3 results in a substantial improvement in the initial lower bound over α_2.

Table 18.14 shows the progression of the upper and lower bounds on the global solution for Formulation I in the case where $\gamma = 0$. An upper bounding problem is solved at each iteration, using the current lower bound solution as a starting point. The upper bound is equal to the global solution at iteration seven. In addition, the initial lower bound is within two percent of the initial upper bound.

Investment Cost Coefficients			Prices of Products	
Stage	α_j	β_j	Product	p_i
1	5	0.6	1	5.5
2	5	0.6	2	7.0
3	5	0.6		

Table 18.10. Cost and Price Data for Example 1

	Size Factors				Processing Times			
		Stage				Stage		
Scenario	Product	1	2	3	Product	1	2	3
1	1	2.5	3.5	4.5	1	7	19	7
	2	4.5	6.5	3.5	2	15	3	3
2	1	1.5	2.5	3.5	1	9	21	9
	2	3.5	5.5	2.5	2	17	5	5
3	1	2	3	4	1	8	20	8
	2	4	6	3	2	16	4	4

Table 18.11. Processing Parameter Data for Example 1

Example 2. This example requires the design of a batch plant producing four products in six stages with one unit per stage. As in the Illustrative Example, only the demands are considered as uncertain parameters. The uncertain demands have the following normal distribution functions, N(150,10), N(150,8), N(180,9), and N(160,10) for products 1, 2, 3, and 4 respectively, and are represented by five quadrature points.

The SPC problem has 649 total constraint equations, of which 625 are nonlinear constraints with 2500 nonlinear terms. The UIS formulation has 3774 total constraints, of which 3750 are nonlinear with 15000 nonlinear terms. There are a total of 2510 variables. The upper and lower bounds on the equipment

γ value	Optimal Profit	Optimal Design				
		V_1	V_2	V_3	B_1	B_2
Single-Product Campaign						
0	876.582	2159	3119	3886	864	480
4	841.932	2285	3300	4112	914	508
8	827.730	2410	3481	4338	964	536
Unlimited Intermediate Storage						
0	1097.265	1509	2113	2716	604	325

Table 18.12. Solution for Example 1

γ	Using α_1			Using α_2			Using α_3		
	Init. LB	# Itrs.	CPU s	Init. LB	# Itrs.	CPU s	Init. LB	# Itrs.	CPU s
Single-Product Campaign									
0	-892.82	13	27.57	-890.86	5	14.55	-882.53	4	11.65
4	-860.24	14	22.30	-857.79	6	13.55	-848.04	5	11.39
8	-843.39	11	18.30	-840.53	4	7.83	-831.61	2	4.48
Unlimited Intermediate Storage									
0	-1103.85	5	40.73	-1103.19	4	29.41	-1100.65	2	18.22

Table 18.13. Computational Results for Example 1

volumes, V_j, were again 4500 and 500. A relative tolerance of 0.003 was used and the horizon time was 8, while the penalty coefficient, γ, was 0. The data are given in Tables 18.15 and 18.16, and the results are shown in Tables 18.17 and 18.18.

As in Example 1, the UIS scheduling formulation results in smaller equipment and batch sizes, which increases the expected profit. A tighter initial lower bound was obtained using α_2 than α_1, and the algorithm converged in one fewer iteration. Due to the size of this problem, more computational effort

Iteration	Upper Bound	Lower Bound	Relative Difference
1	-876.5781766	-892.8253296	0.01851
2	-876.5795104	-887.0662490	0.01195
3	-876.5818400	-887.0645756	0.01194
4	-876.5818403	-886.9900931	0.01186
5	-876.5818403	-885.8041489	0.01050
6	-876.5818403	-882.0419320	0.00622
7	-876.5818419	-880.8482292	0.00486
8	-876.5818419	-880.8463604	0.00485
9	-876.5818419	-879.8040916	0.00367
10	-876.5818419	-879.8022239	0.00366
11	-876.5818419	-879.6526083	0.00349
12	-876.5818419	-879.6408780	0.00348
13	-876.5818419	-879.0291303	0.00278

Table 18.14. Progression of Upper and Lower Bounds for Example 1, $(\alpha_1; \gamma = 0)$

was required to solve the upper and lower bounding problems than in the first two examples. In addition, notice the large increase in CPU time for the UIS formulation. When α_3 was used to construct the lower bounding functions, the lower bound on the global solution was within the specified relative tolerance in the first iteration. This example demonstrates the ability of the method to provide very tight bounds on the global solution in few iterations.

Example 3. In this example, the design of a plant which produces five products in six stages with a different (fixed) number of pieces of equipment per stage is considered. Only the demands are taken to be uncertain parameters and five quadrature points are used for each uncertain demand. This results in a large problem with 15636 variables and 3155 constraints. Of these constraints, 3125 are nonlinear, containing 15625 nonconvex terms.

Only the single-product campaign formulation was used for this example. The demands follow normal distributions with the following forms, N(250,10), N(150,8), N(180,9), N(160,6), and N(120,3). The bounds on the V variables are 500 and 4500, and a relative tolerance of 0.015 was used. The time horizon was

Investment Cost Coefficients		
Stage	α_j	β_j
1	10	0.6
2	10	0.6
3	10	0.6
4	10	0.6
5	10	0.6
6	10	0.6

Prices of Products	
Product	p_i
1	3.5
2	4.0
3	3.0
4	2.0

Table 18.15. Cost and Price Data for Example 2

	Size Factors						Processing Times					
	Stage						Stage					
Product	1	2	3	4	5	6	1	2	3	4	5	6
1	8.0	2.0	5.2	4.9	6.1	4.1	7.0	8.3	6.0	7.0	6.5	8.0
2	0.7	0.8	0.9	3.8	2.1	2.5	6.8	5.0	6.0	4.8	5.5	5.8
3	0.7	2.6	1.6	3.4	3.2	2.9	4.0	5.9	5.0	6.0	5.5	4.5
4	4.7	2.3	1.6	2.7	1.2	2.5	2.4	3.0	3.5	2.5	3.0	2.8

Table 18.16. Processing Parameter Data for Example 2

Optim. Profit	Optimal Design									
	V_1	V_2	V_3	V_4	V_5	V_6	B_1	B_2	B_3	B_4
Single-Product Campaign										
750.184	2875	1407	1869	2385	2192	1569	359	628	541	612
Unlimited Intermediate Storage										
830.338	2703	1323	1757	2045	2061	1475	338	538	509	575

Table 18.17. Solution of Example 2 for $\gamma = 0$

6 and the penalty coefficient was 0. The processing parameters, price, and cost data are given in Tables 18.19 and 18.20, and the results are shown in Tables

Using α_1			Using α_2			Using α_3		
Initial LB	# Itrs.	CPU s	Initial LB	# Itrs.	CPU s	Initial LB	# Itrs.	CPU s
Single-Product Campaign								
-757.24	6	551.56	-756.23	5	471.74	-751.56	1	35.65
Unlimited Intermediate Storage								
-836.85	5	2942.32	-835.90	4	4325.90	-831.71	1	398.18

Table 18.18. Computational Results for Example 2

18.21 and 18.22. Again, the initial lower bound provided by α_3 is the tightest, followed by α_2. In each case, the initial lower bound is extremely close to the global solution, and all formulations converged in the first iteration. Note that the CPU time required for the α_2 formulation was much higher than for the other two α expressions.

Investment Cost Coefficients				Prices of Products	
Stage	α_j	β_j	N_j	Product	p_i
1	0.25	0.6	3	1	3.5
2	0.25	0.6	2	2	4.0
3	0.25	0.6	3	3	3.0
4	0.25	0.6	2	4	2.0
5	0.25	0.6	1	5	4.5
6	0.25	0.6	2		

Table 18.19. Cost and Price Data for Example 3

18.8 COMPARISON TO ALTERNATIVE UNDERESTIMATING APPROACHES

A major advantage of the αBB approach in comparison to underestimation schemes that are based on the methods of Al-Khayyal and Falk (1983), McCormick (1976), as well as products of univariate functions is that it does not require the addition of any new constraints or variables. This advantage becomes quite pronounced as the size of the large-scale optimization problems increases.

	Size Factors						Processing Times					
	Stage						Stage					
Product	1	2	3	4	5	6	1	2	3	4	5	6
1	7.9	2.0	5.2	4.9	6.1	4.2	6.4	4.7	8.3	3.9	2.1	1.2
2	0.7	0.8	0.9	3.4	2.1	2.5	6.8	6.4	6.5	4.4	2.3	3.2
3	0.7	2.6	1.6	3.6	3.2	2.9	1.0	6.3	5.4	11.9	5.7	6.2
4	4.7	2.3	1.6	2.7	1.2	2.5	3.2	3.0	3.5	3.3	2.8	3.4
5	1.2	3.6	2.4	4.5	1.6	2.1	2.1	2.5	4.2	3.6	3.7	2.2

Table 18.20. Processing Parameter Data for Example 3

Optim.	Optimal Design										
Profit	V_1	V_2	V_3	V_4	V_5	V_6	B_1	B_2	B_3	B_4	B_5
3731.08	2789	1901	1836	2460	2187	1982	353	724	683	593	528

Table 18.21. Solution of Example 3 for $\gamma = 0$

Using α_1			Using α_2			Using α_3		
Init. LB	# Iters.	CPU s	Init. LB	# Iters.	CPU s	Init. LB	# Iters.	CPU s
-3731.40	1	1232.7	-3731.25	1	7051.8	-3731.13	1	2089.4

Table 18.22. Computational Results for Example 3

For example, an alternate convex lower bounding scheme is based on representing the nonconvex term as a product of univariate functions, (Maranas and Floudas, 1995). The nonconvex term, $Q_i^{qp} \exp(t_{Li}^p - b_i)$ is a product of two univariate functions $f(Q_i^{qp}) \cdot g(b_i)$, where $f(Q_i^{qp}) = Q_i^{qp}$ and $g(b_i) = \exp(t_{Li}^p - b_i)$. In this approach, a new variable is substituted for each nonconvex term, and convex lower bounds on the substituted variable are constructed. In this case,

$$s_i^{qp} = Q_i^{qp} \exp(t_{Li}^p - b_i)$$

$$s_i^{qp} \geq Q_i^{Lqp} \exp(t_{Li}^p - b_i) + Q_i^{qp} \exp(t_{Li}^p - b_i^U) - Q_i^{Lqp} \exp(t_{Li}^p - b_i^U)$$

$$s_i^{qp} \geq Q_i^{Uqp} \exp(t_{Li}^p - b_i) + Q_i^{qp} \exp(t_{Li}^p - b_i^L) - Q_i^{Uqp} \exp(t_{Li}^p - b_i^L)$$

This approach requires the addition of $N \cdot Q \cdot P$ new variables and $2 \cdot N \cdot Q \cdot P$ new constraints.

The three αBB methods are compared with the univariate functions convex lower bounding scheme in Table 18.23 for three examples. The advantage of the αBB approach becomes apparent as the size of the problem increases. For the illustrative example, and Example 1, the αBB methods provide tighter initial lower bounds than the alternate method. However, since the problems are not especially large, the advantage in CPU time is minimal. Example 2 is much larger than the other two examples, and the advantage of the αBB method is clearly shown. The alternate method took roughly nine times the computational effort of the α_1 method.

Summary

This chapter presented a procedure for finding the globally optimal design of multiproduct and multipurpose batch plants under uncertainty. The problem with a fixed number of equipment per stage was formulated as a single nonlinear optimization problem. The uncertain demands were represented by a Gaussian quadrature formulation, and the uncertain processing parameters were handled through a scenario-based approach. The design problem was formulated for both single-product campaign and mixed-product campaign with unlimited intermediate storage scheduling strategies. A key theoretical property has been developed, which is an analytical expression for the minimum value of α needed to form a convex lower bound of the nonconvex constraints. This property allows very tight bounds on the global solution to be generated in a small number of iterations. The αBB algorithm with a modified branching criterion was used to converge to the global solution. Several example problems were presented, which demonstrate the effectiveness of the proposed approach to large-scale multiproduct batch design problems under uncertainty. Additional computational studies and the analysis of the global optimization approach for multipurpose batch plants can be found in Harding and Floudas (1997). Global optimization test problems on multiproduct and multipurpose batch design under uncertainty can also be found in the recent book by Floudas *et al.* (1999).

α Approach	Init. LB	# Iters	CPU s	# Variables	# Constraints
Illustrative Example: $\gamma = 0$					
α_1	-987.840	4	1.30	55	31
α_2	-986.777	3	1.47	55	31
α_3	-982.180	2	0.75	55	31
Univariate	-990.076	10	2.01	105	131
Example 1: $\gamma = 0$					
α_1	-892.825	13	27.57	155	93
α_2	-890.861	5	14.55	155	93
α_3	-882.530	4	11.65	155	93
Univariate	-900.901	10	14.38	305	393
Example 2					
α_1	-757.247	6	551.56	2510	649
α_2	-756.238	5	471.74	2510	649
α_3	-751.566	1	35.65	2510	649
Univariate	-796.887	8	4825.96	5010	5649

Table 18.23. Comparison of Lower Bounding Strategies

19 THE αBB APPROACH IN PARAMETER ESTIMATION

In this chapter, we present a global optimization framework for the parameter estimation of nonlinear algrebraic models through the *error-in-variables* approach. Section 19.1 provides the motivation and reviews the previous research contributions. Section 19.2 introduces the basics of the maximum likelihood parameter estimation. Section 19.3 presents the global optimization approach which is based on the general principles of the αBB with a number of modifications. Section 19.4 describes the detailed algorithmic steps of the global optimization approach as it is applied to the parameter estimation problem. Section 19.5 presents representative computational studies and comparisons with interval analysis based approaches. The presented material in this chapter is based on the work of Esposito and Floudas (1998).

19.1 INTRODUCTION

Mathematical models which accurately predict physical phenomena are essential in many fields of engineering and applied sciences. These models frequently contain adjustable parameters which need to be determined from available experimental data. In the chemical engineering field, mathematical models form the basis for the design, control, and optimization of process systems. The use of nonlinear models, many of which are implicit in nature, introduces an added level of complexity into the numerical estimation of the model parameters.

C. A. Floudas, *Deterministic Global Optimization*
© Springer Science+Business Media Dordrecht 2000

Many different statistical methods exist for the estimation of the parameters in both linear and nonlinear models, as shown by Bard (1974). In particular, the maximum likelihood method, has been presented extensively in the literature. In the maximum likelihood approach, errors in all measured variables, rather than just in a set of dependent variables, are accounted for, thus leading to unbiased estimates of the true parameter values. One formulation of this approach, which has become popular, assumes that the error is normally distributed with a zero mean and known diagonal covariance matrix. The resulting optimization problem is referred to as the *error-in-variables* formulation. Two major difficulties arise in addressing this problem. First, since error is accounted for in all variables, the optimization is performed not only over the parameters, but also over a fitted set of data variables thus leading to many degrees of freedom. Second, due to the nonlinear nature of the models, the resulting optimization problem is nonconvex and may contain many local minima in the area of interest.

Over the years many different methods have been proposed to address this formulation. Bard and Lapidus (1968) provided a description of the early work in the area as it relates to the estimation of kinetic parameters in complex reaction networks. Southwell (1969) presented a method also based on a gradient approach, but through the use of iterative matrix calculations. Schwetlick and Tiller (1985) extended this approach to include models which have more than two variables and are highly nonlinear in nature. They accomplished this by exploiting the structure of the Jacobian matrix, and developing a different correction calculation to overcome difficulties in the Gauss-Newton correction for strongly nonlinear problems. This correction is based on a regularization proposed by Marquardt (1963) for the least squares problem. Britt and Luecke (1973) derived an algorithm based on the use of Lagrange multipliers for the model equations which enter into the problem as equality constraints. Their method involves the use of successive constraint linearization and closely mimics the Gauss-Newton approach for the least squares problem. Fariss and Law (1979) presented an approach which modifies the form of the error response to allow for the detection of gross errors in the data. Their numerical method employs a gradient based approach, but only requires analytical first derivatives. Fabries and Renon (1975) developed a method specifically for thermodynamic models also using a Gauss-Newton approach. Anderson *et al.* (1978) provided an extension of this work which incorporates two equation models to fully describe the system. They used a successive linearization approach in the determination of the model parameters.

Patino-Leal and Reilly (1982) addressed the vapor-liquid equilibrium problem. Through the linearization of the model equations, and integration over the fitted data sets, they formulated an optimization problem which contains only the parameters and linearization points. Valko and Vajda (1987) presented an almost identical approach with the exception that the outer optimization problem is reformulated so that a Gauss-Marquardt method can be directly applied. In the approach by Rod and Hancil (1980) instead of using a linearization of

the model equations in calculating the values of the fitted data variables in the inner problem, an optimization problem is solved.

Another approach to reduce the size of the variable set is via problem decomposition that allows for the optimization to be performed only in the space of the parameters. Dovi and Paladino (1989) accomplished this through the use of constrained variation in which the change in one variable can be written as a function of the change in another. Tjoa and Biegler (1992) proposed a method based on successive quadratic programming. At each quadratic subproblem, the optimality conditions in terms of the fitted data variables are solved analytically and included in the optimization.

Reviews and comparisons of the various methods are given by Ricker (1984), Stewart *et al.* (1992), and Kim *et al.* (1990).

All the aforementioned approaches address the first of the two difficulties referred to earlier. They only offer convergence to a local solution of a formulation which may have multiple minima. The result could be erroneous parameter estimates and inadequate models. The approach of Esposito and Floudas (1998) is the first one that guarantees convergence to the global minimum and exploits the structure of the mathematical formulation. Moore *et al.* (1992) and Csendes and Ratz (1995) both provide global optimization algorithms based on interval arithmetic. As illustrations of their techniques, parameter estimation problems, in the form of nonlinear, unconstrained least squares problems were solved.

In this chapter, a deterministic global optimization algorithm based on a branch and bound framework is proposed to address the error-in-variables formulation with the mathematical structure of the model being twice continuously differentiable.

19.2 MAXIMUM LIKELIHOOD ESTIMATION

The model takes the form of an implicit algebraic system of equations

$$\mathbf{f}(\theta, \mathbf{z}) = \mathbf{0} \tag{19.1}$$

where θ is a vector of p unknown parameters, $\mathbf{z}$ is a vector of n experimentally measured variables, and $\mathbf{f}$ represents the system of l algebraic functions.

All experimentally measured variables are affected to some extent by error. The measurements are related to the true values through:

$$\mathbf{z}_\mu = \zeta_\mu + \mathbf{e}_\mu \quad \mu = 1, \ldots, m \tag{19.2}$$

where ζ_μ is vector of unknown true values of the experimentally measured variables, $\mathbf{z}_\mu$, at the μ data point, and $\mathbf{e}_\mu$ is the vector of additive error.

We define the likelihood function (L) as the probability of the observed errors in all data points occurring given a set of statistical parameters (ψ) for the distribution used to represent those errors. The errors are functions of the selected values for the parameters through the model equations

$$L(\theta, \psi) \equiv p\,(\mathbf{E}(\theta) \,|\, \psi) \tag{19.3}$$

with $\mathbf{E} \equiv [\mathbf{e}_1^T, \mathbf{e}_2^T, \ldots, \mathbf{e}_m^T]^T$. As it is frequently assumed, the measurement errors from different data points (e.g., experiments) are considered as uncorrelated or independent. Then, the joint probability density shown in (19.3) can be expressed as a product of the individual probability densities, and hence the likelihood function becomes:

$$L(\theta, \psi) = \prod_{\mu=1}^{m} p\left(\mathbf{e}_\mu(\theta) \,|\, \psi_\mu\right). \tag{19.4}$$

The maximum likelihood estimator seeks to determine values for θ and ψ which maximize L. For convenience it is much easier to work with $(\ln L)$. Maximization of $(\ln L)$ is equivalent to the maximization of the original function L. This results in:

$$\ln L(\theta, \psi) = \sum_{\mu=1}^{m} \ln p\left(\mathbf{e}_\mu(\theta) \,|\, \psi_\mu\right). \tag{19.5}$$

If we assume that the errors follow a normal distribution with zero mean and covariance matrix $\mathbf{V}$, we have:

$$p(\mathbf{e}|\mathbf{V}) = N(\mathbf{e}, \mathbf{V}) = \frac{2\,\pi^{-\mathbf{n}/2}}{\sqrt{|\mathbf{V}|}} \exp\left[-\frac{1}{2}\,\mathbf{e}^{\mathbf{T}}\,\mathbf{V}^{-1}\,\mathbf{e}\right]. \tag{19.6}$$

Substituting (19.6) into (19.5), we obtain:

$$\ln L = -\frac{n\,m}{2}\ln 2\pi - \frac{1}{2}\sum_{\mu=1}^{m}\ln|\mathbf{V}_\mu| - \frac{1}{2}\sum_{\mu=1}^{m} e_\mu^T\,\mathbf{V}_\mu^{-1}\,e_\mu. \tag{19.7}$$

Different forms of (19.7) exist depending on what is known, or assumed about the covariance matrix. In the sequel, we outline three common cases:

Case 1: If the covariance matrix is known completely for each experiment, then only the final term is variable. Therefore maximization of $\ln L$ is equivalent to:

$$\min \sum_{\mu=1}^{m} \mathbf{e}_\mu^T\,\mathbf{V}_\mu^{-1}\,\mathbf{e}_\mu. \tag{19.8}$$

Case 2: If the covariance matrix is assumed to be known and to be the same in each experiment, that is, $\mathbf{V}_1 = \mathbf{V}_2 = \cdots = \mathbf{V}$, then (19.8) reduces to:

$$\min \sum_{\mu=1}^{m} \mathbf{e}_\mu^T \mathbf{V}^{-1} \mathbf{e}_\mu. \tag{19.9}$$

Case 3: If it is assumed that the errors in each experiment are known, equal, and independent, that is, the covariance matrix is diagonal with elements v_i, then (19.9) can be written as:

$$\min \sum_{\mu=1}^{m} \sum_{i=1}^{n} e_{\mu,i}^2 \, v_i^{-1} \tag{19.10}$$

where $e_{\mu,i}$ is the ith component on the vector $\mathbf{e}_\mu$, that is, the error associated with the ith variable in the μth experiment.

This last form is the one that we will study in this chapter. Defining v_i as σ_i^2 where σ_i is the standard deviation assigned to the ith variable in all experiments, substituting the definition of $e_{\mu,i}$ from (19.2), and subjecting the minimization to the model equations, we have the following optimization problem, known as the *error-in-variables* formulation:

$$\min_{\hat{\mathbf{z}}_\mu \, \theta} \sum_{\mu=1}^{r} \sum_{i=1}^{m} \frac{(\hat{z}_{\mu,i} - z_{\mu,i})^2}{\sigma_i^2} \tag{19.11}$$

s.t.

$$\mathbf{f}(\hat{\mathbf{z}}_\mu, \theta) = \mathbf{0} \quad \mu = 1 \ldots m.$$

The optimization variables, $\hat{\mathbf{z}}_\mu$, in (19.11) are referred to as the fitted data variables. These are approximations, obtained through the optimization, of the true values of the experimental data, ζ_μ.

Remark 19.2.1 This formulation has the following mathematical properties:

(i) For a known and constant experimental variance the objective function is convex.

(ii) Since the model equations enter into the formulation as equality constraints, nonlinear models result in nonconvex optimization problems.

(iii) Since the minimization is not only over the parameters, but also over the fitted data variables, $\hat{\mathbf{z}}_\mu$, even models linear in the parameters, such as $z_2 = a z_1 + b$, are nonlinear due to the bilinear term $a z_1$.

(iv) A group of constraints, corresponding to the model equations, are written for each data point, $\mu = 1 \ldots m$. This results in the fitted data variables, $\hat{\mathbf{z}}_\mu$, appearing in one set of the constraints, while the parameters, θ, appear in every set.

(v) An unconstrained formulation is sometimes possible through substitution of the model equations into the objective function.

These mathematical properties form the basis for the development of a global optimization approach for the *error-in-variables* formulation.

19.3 GLOBAL OPTIMIZATION FRAMEWORK

In order to solve (19.11) to global optimality, the αBB deterministic global optimization algorithm that we discussed in Chapters 11 and 12 (Androulakis *et al.*, 1995; Adjiman and Floudas, 1996; Adjiman *et al.*, 1996; Adjiman *et al.*, 1998a; Adjiman *et al.*, 1998b), was modified and applied to the parameter estimation of nonlinear algebraic models.

The αBB global optimization method guarantees convergence to the global minimum for general twice continuously differentiable constrained and unconstrained NLPs. This is accomplished through the generation of a non-decreasing sequence of lower and a non-increasing sequence of updated upper bounds on the global solution. Finite ϵ-convergence to the global minimum is achieved through the successive subdivision of the region at each level in the branch and bound tree. The sequence of upper bounds on the global solution is obtained by solving, to local optimality, the full nonconvex problem from different starting points. The lower bounds are generated by solving a convex relaxation of the original problem. The underestimation problem is generated by replacing each nonconvex term in the objective function and the constraints by valid underestimating functions as discussed in Chapters 11 and 12.

19.3.1 Important Modifications of αBB

There are three main areas which have a major effect on the convergence rate of the global optimization algorithm. These are: (i) the problem formulation, (ii) the initial bounds on the variables, and (iii) the selection of the branching variable. Each of these three areas was extensively studied so as to develop an efficient algorithm tailored specifically to the parameter estimation of nonlinear algebraic models.

Problem Formulation. The formulation of the problem has a major effect on the quality of the convex relaxation. The goal is to produce a formulation in which the nonconvexities are in as simple form as possible which will allow for the analytical determination of tight convex relaxations. This is accomplished through the use of variable substitutions and model transformations to allow for the use of as many special terms as possible. Another consideration is the number of variables which participate in the nonconvex terms. The fewer variables that are involved in the relaxation, the fewer variables need to be branched on to achieve convergence.

An unconstrained formulation is possible in most cases. Local optimization methods are able to solve these formulations with reasonable computational effort. However, in the framework of this global optimization method uncon-

strained formulations are more difficult to solve. By substituting the nonlinear model equations into an already quadratic objective function, the terms generated are much more complex and involve many variables. This leads to difficulties in obtaining tight convex relaxations for these functions. The effects of various formulations on the convergence of the algorithm is shown extensively through the examples presented in section 19.5.

Variable Bounds. Since the variable bounds appear explicitly in the formulation of the convex underestimators, tighter initial bounds produce tighter initial underestimators, which in turn lead to faster convergence of the algorithm. The objective is to generate sufficiently tight initial bounds to obtain convergence to the global solution in a reasonable amount of time, but not to exclude a physically significant global minimum for the problem. This formulation contains two distinct classes of variables, the fitted data variables, $\hat{\mathbf{z}}_\mu$, and the parameter variables, θ. Since each type of variable has a different effect on the problem, alternative methods are used to generate the initial bounds.

Data set variable bounds: Valid bounds on the fitted data variables can be found from statistical considerations. For a known experimental standard deviation, σ, there is a 99.7% probability that the true value will be within $\pm 3\sigma$ of the observations. Frequently, variables also have absolute values dictated by physical limitations, for example, 0 to 1 for mole fractions. For some problems these bounds are reasonably tight.

A method was developed to generate the tightest possible bounds, while still reasonably ensuring that the global minimum is obtained (Esposito and Floudas, 1998). The first step is to identify a set of bounds on $\hat{\mathbf{z}}_\mu$ which provide an interior local solution ($\hat{\mathbf{z}}_\mu^L < \hat{\mathbf{z}}_\mu^{sol} < \hat{\mathbf{z}}_\mu^U$) and then solve the problem to global optimality using these bounds. This does not ensure that the global solution for the full statistically possible region is identified. Therefore, after the problem is solved, the bounds on the fitted data variables are expanded and the problem is re-solved to global optimality using information obtained from the previous global solution. In the limit as the bounds are expanded to the full statistical region, the absolute global solution will be identified. This method only needs to be applied to those data set variables which participate in nonconvex terms. The other variables are given bounds corresponding to the full statistical region.

Parameter variable bounds: Bounds on the parameter variables are obtained very differently. Loose initial bounds are set, for instance, from physical considerations and prior knowledge. Upon determination of the bounds on the data variables at each outer iteration, the bounds on the parameters are then tightened by solving a series of minimization and maximization problems of the following form:

$$\theta_j^{l,*}/\theta_j^{u,*} = \begin{cases} \min_{\theta\ \hat{\mathbf{z}}_\mu} / \max_{\theta\ \hat{\mathbf{z}}_\mu} & \theta_j \\ \text{subject to} & \mathcal{L}(\theta, \hat{\mathbf{z}}_\mu) \leq 0 \\ & \hat{\mathbf{z}}_\mu^L \leq \hat{\mathbf{z}}_\mu \leq \hat{\mathbf{z}}_\mu^U \\ & \theta^L \leq \theta \leq \theta^U \end{cases} \tag{19.12}$$

where $\mathcal{L}(\theta, \hat{\mathbf{z}}_\mu)$ is the convex underestimation of the model equations.

These problems are solved in sequence, $j = 1, \ldots, p$, where p is the total number of parameters. Since the bounds derived from (19.12) are dependent on the bounds of the other variables, those solved for first are not as tight as those determined last. The sequence is solved repeatedly until the bounds on all the variables do not change significantly.

Branching Methods. In the formulation, the parameter variables appear in every set of constraints, while the fitted data variables appear in only one. Therefore by branching initially on the parameters, the underestimation of the problem is improved the most. At some point though, the fitted data variables will need to be branched on to achieve final convergence. The proposed approach which accomplishes this most efficiently, examines the sum of the deviations caused by each variable in all the nonconvex terms of the problem. These deviations are defined as follows: For a univariate concave term, $f_k(x_i)$, the deviation, $\delta_i^{UT_k}$, of the ith variable is defined as:

$$\delta_i^{UT_k} = f_k(x_i^{sol}) - \left\{ f_k(x_i^l) + \frac{f_k(x_i^u) - f_k(x_i^l)}{x_i^u - x_i^l} (x_i^{sol} - x^l) \right\} \tag{19.13}$$

where x_i^{sol} represents the value of the ith variable at the solution to the lower bounding problem in the current region. For the kth bilinear term, $x_i x_j$, the deviation for the ith variable, $\delta_i^{BT_k}$, includes a weighting by a region reduction measure, and is defined as:

$$\delta_i^{BT_k} = \left| w_k^{sol} - x_i^{sol} x_j^{sol} \right| \left(\frac{x_i^u - x_i^l}{x_i^{u,orig} - x_i^{l,orig}} \right) \tag{19.14}$$

where w_k^{sol} is the value of the kth auxiliary variable at the solution to the lower problem, and $x_i^{l,orig}$ and $x_i^{u,orig}$ are the original lower and upper bounds on the variable x_i. For a general nonconvex term, T_k, the deviation, $\delta_i^{T_k}$, of the i variable is defined as:

$$\delta_i^{T_k} = T_k(\mathbf{x}_T^{sol}) - \mathcal{L}(\mathbf{x}_T^{sol}) \tag{19.15}$$

where $\mathbf{x}_T^{sol}$ is the value of the vector of participating variables at the solution to the lower bounding problem, $T_k(\mathbf{x}_T^{sol})$ is the value of the original nonconvex

term at this solution, and $\mathcal{L}(\mathbf{x}_T^{sol})$ is the value of the underestimating function at the solution. The total deviation, δ_i, for variable i is defined as:

$$\delta_i = \left\{ \sum_{k \in T_i} \delta_i^{T_k} + \sum_{k \in B_i} \delta_i^{BT_k} + \sum_{k \in U_i} \delta_i^{UT_k} \right\} ncon_i \tag{19.16}$$

where B_i, U_i, and T_i are the sets of bilinear, univariate concave, and general terms, that the ith variable participates in, and $ncon_i$ is the number of constraints in the original problem in which the ith variable appears. The branching variable index i^* is selected by:

$$i^* = \arg\max_i \delta_i. \tag{19.17}$$

Other possible branching methods are presented in Chapters 11 and 12, and in Adjiman *et al.* (1998b).

19.4 GLOBAL OPTIMIZATION ALGORITHM

The proposed global optimization approach for the parameter estimation of nonlinear algebraic models consists of the following steps:

STEP 1. Initialization

1. Formulate the problem as a constrained minimization using the guidelines discussed in section 19.3.1.
2. Set the relative ϵ^{rel} or the absolute ϵ^{abs} convergence tolerance for αBB.
3. Set the outer iteration counter $iter^{out} = 1$.
4. Set initial bounds on the parameter variables from physically significant or prior knowledge,

$$\theta^{l,orig} \leq \theta \leq \theta^{u,orig}. \tag{19.18}$$

For fitted data variables which participate in nonconvex terms, set the bounds:

$$\mathbf{z}_\mu - \sigma \leq \hat{\mathbf{z}}_\mu \leq \mathbf{z}_\mu + \sigma \tag{19.19}$$

where σ is the vector of standard deviations, and $\mathbf{z}_\mu$ is the observed values of the data variables. Variables not participating in nonconvex terms have their bounds set at

$$\mathbf{z}_\mu - 3\sigma \leq \hat{\mathbf{z}}_\mu \leq \mathbf{z}_\mu + 3\sigma. \tag{19.20}$$

STEP 2. Determine initial bounds on the data set variables, $\hat{\mathbf{z}}_\mu$. This is accomplished by solving the full nonconvex problem locally. Three cases are identified:

- *The problem is infeasible* - Expand the upper and lower bounds on all the variables by a given amount, δ, and re-solve the problem

$$\hat{\mathbf{z}}_\mu^u = \hat{\mathbf{z}}_\mu^u + \delta \tag{19.21}$$

$$\hat{\mathbf{z}}_\mu^l = \hat{\mathbf{z}}_\mu^l - \delta. \tag{19.22}$$

- *Feasible, but does not have an interior solution* - Expand the active bounds by δ and re-solve.

$$\text{if } \hat{z}_{\mu,i}^{sol} = \hat{z}_{\mu,i}^u \text{ then } \hat{z}_{\mu,i}^u = \hat{z}_{\mu,i}^u + \delta_i \tag{19.23}$$

$$\text{if } \hat{z}_{\mu,i}^{sol} = \hat{z}_{\mu,i}^l \text{ then } \hat{z}_{\mu,i}^l = \hat{z}_{\mu,i}^l - \delta_i. \tag{19.24}$$

- *Has an interior solution* - Stop solving the problem locally.

STEP 3. Tighten the bounds on the parameter variables.

1. Solve (19.12) for each parameter variable.
2. Calculate the total amount the bounds have changed

$$\Delta_B = \sum_{i=1}^{p} \left(1 - \frac{\theta_i^{new,u} - \theta_i^{new,l}}{\theta_i^{old,u} - \theta_i^{old,l}} \right) \tag{19.25}$$

   where the superscript *old* refers to the bounds determined at the previous iteration, and *new* refers to those determined in this iteration.

3. If $\Delta_B \leq 0.10$ then terminate and start the αBB solution.
   Otherwise, set $\theta^{new,l} \rightarrow \theta^{old,l}$; $\theta^{new,u} \rightarrow \theta^{old,u}$ and return to STEP 1.

STEP 4. Solve the problem to global optimality using the αBB algorithm.

1. Initialize the iteration counter $iter^{\alpha\text{BB}} = 1$.
2. Solve the full nonconvex problem locally to generate an upper bound (UB) on the global solution and store the variable values.

$$obj^* \rightarrow UB \ , \ \hat{\mathbf{z}}_\mu^* \rightarrow \hat{\mathbf{z}}_\mu^{UB} \ , \ \theta^* \rightarrow \theta^{UB}.$$

3. Solve the full convex relaxation locally to generate a lower bound (LB) on the global solution. Store the objective value and values of all variables and the α's.

$$obj^* \rightarrow obj^{1,1} \ , \ \hat{\mathbf{z}}_\mu^* \rightarrow \hat{\mathbf{z}}_\mu^{1,1} \ , \ \theta^* \rightarrow \theta^{1,1} \ , \ \mathbf{w}^* \rightarrow \mathbf{w}^{1,1} \ , \ \alpha \rightarrow \alpha^{1,1}.$$

4. If $UB - LB \leq \epsilon^{abs}$ or $\frac{UB - LB}{UB} \leq \epsilon^{rel}$, then terminate the αBB algorithm.

5. If desired, update the bounds on selected variables by solving the following min/max problems

$$x_j^{l,*}/x_j^{u,*} = \begin{cases} \min_{\theta\ \hat{\mathbf{z}}_\mu} / \max_{\theta\ \hat{\mathbf{z}}_\mu} & x \\ \text{subject to} & \mathcal{L}(\theta, \hat{\mathbf{z}}_\mu) \leq 0 \\ & \hat{\mathbf{z}}_\mu^L \leq \hat{\mathbf{z}}_\mu \leq \hat{\mathbf{z}}_\mu^U \\ & \theta^L \leq \theta \leq \theta^U \end{cases} \tag{19.26}$$

   where $\boldsymbol{x}$ represents the vector $[\hat{\mathbf{z}}_\mu\ \theta]$ and $j \in J$, where J is the set of variables selected for updating.

6. Select the variable to branch the domain on using the expressions developed in section 19.3.1.

7. Divide the region by bisecting along the selected branching variable.

8. If desired, update the bounds on selected variables in each new region by solving (19.26).

9. Solve the upper and lower bounding problems to local optimality in each newly created region ($r = 1, 2$). Update the upper bound if a local solution in one of the new regions is less than the current upper bound:

   if $obj^{upper} < UB$ then

$$obj^{upper} \rightarrow UB\ ,\quad \hat{\mathbf{z}}_\mu^* \rightarrow \hat{\mathbf{z}}_\mu^{UB}\ ,\quad \theta^* \rightarrow \theta^{UB}$$

   Store the solutions to the lower problems that are less than the current upper bound:

   if $obj^{lower} < UB$ then

$$obj^{lower} \rightarrow obj^{iter,r}\ , \hat{\mathbf{z}}_\mu^* \rightarrow \hat{\mathbf{z}}_\mu^{iter,r}\ , \theta^* \rightarrow \theta^{iter,r}\ , \mathbf{w}^* \rightarrow \mathbf{w}^{iter,r}, \alpha \rightarrow \alpha^{iter,r}.$$

10. Select the region with the lowest lower bounding solution. This becomes the new lower bound:

$$LB = \min_{iter\ r} obj^{iter,r}. \tag{19.27}$$

   Remove this region from the list of stored regions.

11. Increment the iteration counter $iter^{\alpha\text{BB}} = iter^{\alpha\text{BB}} + 1$ and return to STEP 4.

STEP 5. Check for global convergence:

if $iter^{out} = 1$ or $sol^{iter^{out}} \neq sol^{iter^{out}-1}$ then

$$\hat{\mathbf{z}}_\mu^u = \hat{\mathbf{z}}_\mu^u + \gamma\left(\hat{\mathbf{z}}_\mu^u - \hat{\mathbf{z}}_\mu^l\right) \ , \ \hat{\mathbf{z}}_\mu^l = \hat{\mathbf{z}}_\mu^l - \gamma\left(\hat{\mathbf{z}}_\mu^u - \hat{\mathbf{z}}_\mu^l\right)$$

$$\theta^u = \theta^{u,orig} \ , \ \theta^l = \theta^{l,orig}$$

$$iter^{out} = iter^{out} + 1 \text{ and return to STEP 2.}$$

where $sol^{iter^{out}}$ is the global solution determined by αBB in the $iter^{out}$ outer iteration. Otherwise terminate.

The algorithm is presented in a flowsheet form in Figure 19.1. It should be noted that the convergence criterion given in step 5 guarantees convergence to the global minimum in the limit as the parameter γ is increased such that the bounds are expanded to the point in which the full statistical region is enclosed. For practical purposes a value of 0.1 was used in all the examples and has resulted in convergence to the global minimum in every case tested, even in those which the first outer iteration resulted in a local minimum. This algorithm has been implemented in a C program as an extension to the original αBB, where MINOS 5.4 (Murtagh and Saunders, 1983) is used to perform the needed local optimizations.

19.5 COMPUTATIONAL STUDIES

The following three examples illustrate the important issues of (i) problem formulation; (ii) types of underestimators; (iii) unconstrained vs. constrained optimization; (iv) tightness of different underestimators; and (v) computational performance and convergence characteristics. All problems were solved on a Hewlett Packard C160.

19.5.1 Example 1

This example is taken from Tjoa and Biegler (1992) and Rod and Hancil (1980). The model equation takes the form of:

$$z_2 = \theta_1 + \frac{1}{z_1 - \theta_2}. \tag{19.28}$$

Data sets of 25, 50, 75, and 100 points were generated for this problem using values of $\theta_1 = 2.00$ and $\theta_2 = 6.00$. The values for z_1 ranged from 0 to 3.0. Random errors from a uniform distribution [-0.025, 0.025] were added to both z_1 and z_2. An example of a data set of 100 points along with the model equation used to generate the set is shown in Figure 19.2.

This example illustrates the effect of problem formulation on the performance of the algorithm since three different formulations are studied.

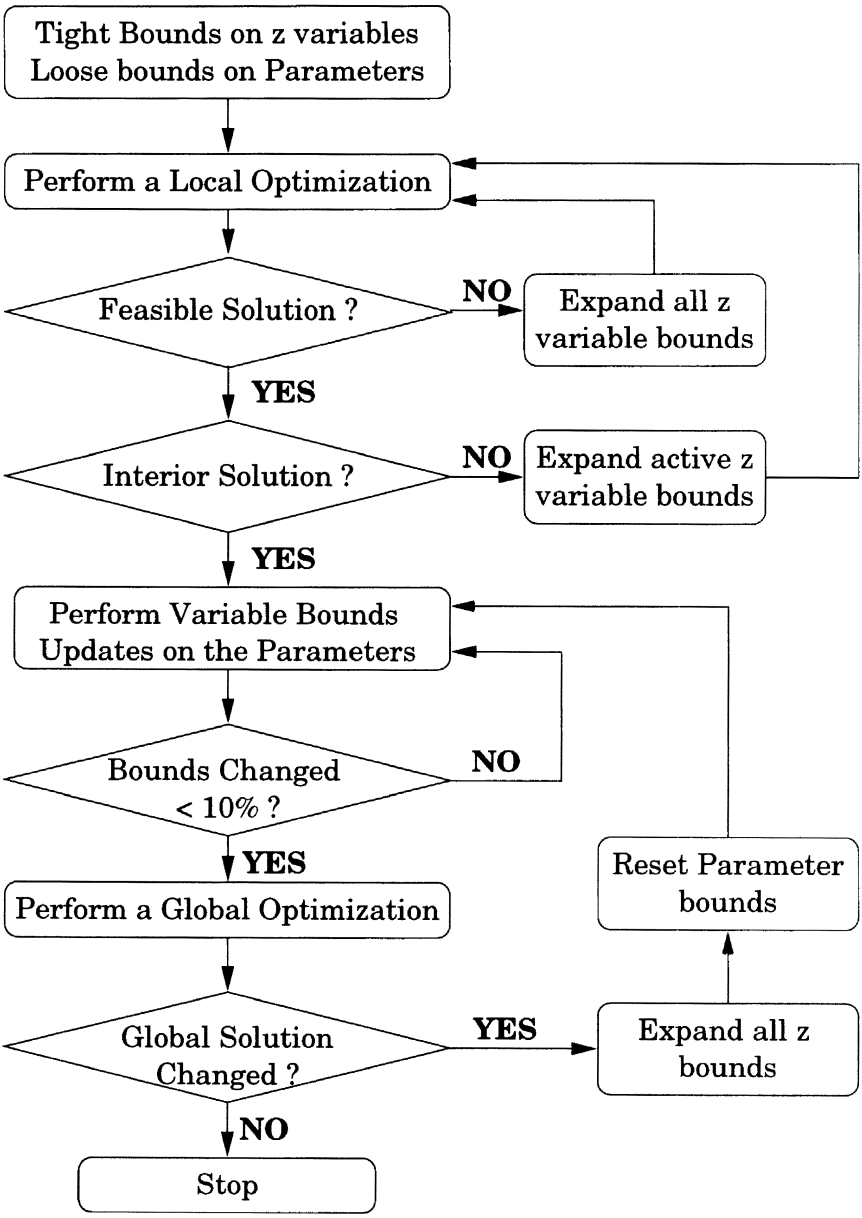


Figure 19.1. Flowsheet of global optimization approach for parameter estimation

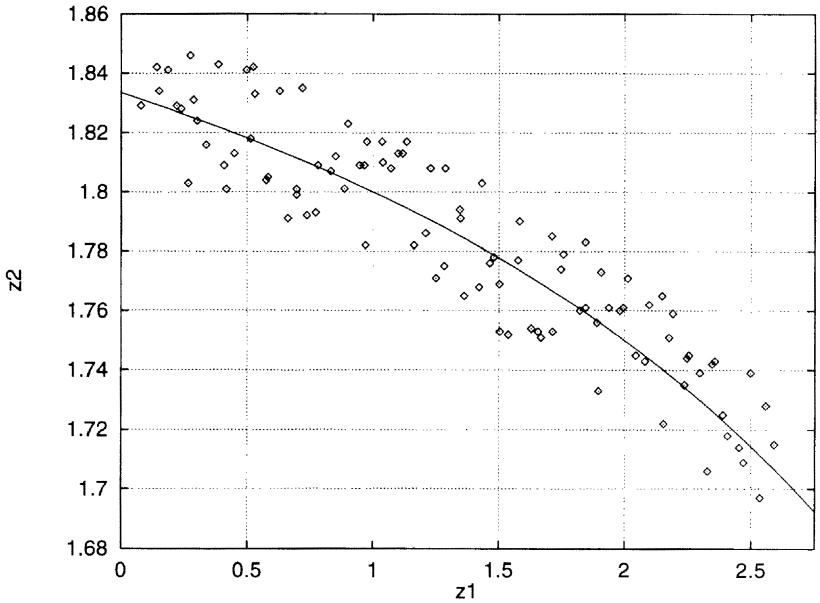


Figure 19.2. Sample 100 point data set for Example 1 and the true model equation (–)

Formulation 1. The first formulation simply has the model equation as a constraint in the optimization and will be referred to as the α formulation:

$$\min_{\hat{\mathbf{z}}_\mu\,\theta} \sum_{\mu=1}^{m} \sum_{i=1}^{2} (\hat{z}_{\mu,i} - z_{\mu,i})^2$$

$$\text{s. t.} \tag{19.29}$$

$$\begin{aligned} -\hat{z}_{\mu,2} + \theta_1 + \frac{1}{\hat{z}_{\mu,1} - \theta_2} &\leq 0 \\ \hat{z}_{\mu,2} - \theta_1 - \frac{1}{\hat{z}_{\mu,1} - \theta_2} &\leq 0. \end{aligned}$$

If the relationship $\hat{z}^u_{\mu,1} < \theta^l_2$ holds (as it does given the data sets generated), then the second constraint is convex. The term in the first constraint is treated as a general nonconvex term. The resulting convex relaxation is:

$$
\begin{aligned}
\min_{\hat{\mathbf{z}}_\mu\,\theta} \quad & \sum_{\mu=1}^{m} \sum_{i=1}^{2} (\hat{z}_{\mu,i} - z_{\mu,i})^2 \\
\text{s. t.} \quad & \\
- \quad & \hat{z}_{\mu,2} + \theta_1 + \frac{1}{\hat{z}_{\mu,1} - \theta_2} \\
+ \quad & \alpha_\mu \left\{ \left(\hat{z}^u_{\mu,1} - \hat{z}_{\mu,1}\right)\left(\hat{z}^l_{\mu,1} - \hat{z}_{\mu,1}\right) + \left(\theta^u_2 - \theta_2\right)\left(\theta^l_2 - \theta_2\right) \right\} \leq 0 \\
& \hat{z}_{\mu,2} - \theta_1 - \frac{1}{\hat{z}_{\mu,1} - \theta_2} \leq 0.
\end{aligned}
\tag{19.30}
$$

The expression for the value of α_μ takes the form :

$$
\alpha_\mu = \frac{2}{(\theta^l_2 - \hat{z}^u_{\mu,1})^3}. \tag{19.31}
$$

Formulation 2. A second constrained formulation is possible in which all the nonconvexities take the form of bilinear terms.

$$
\begin{aligned}
\min_{\hat{\mathbf{z}}_\mu\,\theta} \quad & \sum_{\mu=1}^{m} \sum_{i=1}^{2} (\hat{z}_{\mu,i} - z_{\mu,i})^2 \\
\text{s. t.} \quad & \\
& -\hat{z}_{\mu,1}\,\hat{z}_{\mu,2} + \hat{z}_{\mu,2}\,\theta_2 + \hat{z}_{\mu,1}\,\theta_1 - \theta_1\,\theta_2 = 1.
\end{aligned}
\tag{19.32}
$$

The resulting convex relaxation is:

$$
\begin{aligned}
\min_{\hat{\mathbf{z}}_\mu\,\theta} \quad & \sum_{\mu=1}^{m} \sum_{i=1}^{2} (\hat{z}_{\mu,i} - z_{\mu,i})^2 \\
\text{s. t.} \quad & \\
& -w_{\mu,1} + w_{\mu,2} + w_{\mu,3} - w_4 = 1 \\
& w_{\mu,1} \rightarrow \hat{z}_{\mu,1}\,\hat{z}_{\mu,2} \\
& w_{\mu,2} \rightarrow \hat{z}_{\mu,2}\,\theta_2 \\
& w_{\mu,3} \rightarrow \hat{z}_{\mu,1}\,\theta_1 \\
& w_4 \rightarrow \theta_1\,\theta_2.
\end{aligned}
\tag{19.33}
$$

Formulation 3. Finally an unconstrained formulation can be generated by substituting the model equation directly into the objective function. The formulation is:

$$
\min_{\hat{\mathbf{z}}_\mu\,\theta} \sum_{\mu=1}^{m} (z_{\mu,1} - \hat{z}_{\mu,1})^2 + \left(\theta_1 + \frac{1}{\hat{z}_{\mu,1} - \theta_2} - z_{\mu,2}\right)^2. \tag{19.34}
$$

Expanding (19.34) and grouping the terms results in:

$$\begin{aligned}\min_{\hat{\mathbf{z}}_\mu\,\theta} \quad & -\sum_{\mu=1}^{m} 2\,z_{\mu,2}\,\theta_1 + 2\,z_{\mu,1}\,\hat{z}_{\mu,1} \\ & + \sum_{\mu=1}^{m} z_{\mu,2}^2 + z_{\mu,1}^2 + \hat{z}_{\mu,1}^2 + \theta_1^2 + \frac{1}{(\hat{z}_{\mu,1} - \theta_2)^2} \\ & + \sum_{\mu=1}^{m} \frac{2\,\theta_1 - 2\,z_{\mu,2}}{\hat{z}_{\mu,1} - \theta_2}.\end{aligned} \tag{19.35}$$

In formulation (19.35) only the last term is nonconvex and treated as a term of general structure. The resulting convex relaxation is:

$$\begin{aligned}\min_{\hat{\mathbf{z}}_\mu\,\theta} \quad & -\sum_{\mu=1}^{m} 2\,z_{\mu,2}\,\theta_1 + 2\,z_{\mu,1}\,\hat{z}_{\mu,1} \\ & + \sum_{\mu=1}^{m} z_{\mu,2}^2 + z_{\mu,1}^2 + \hat{z}_{\mu,1}^2 + \theta_1^2 + \frac{1}{(\hat{z}_{\mu,1} - \theta_2)^2} \\ & + \sum_{\mu=1}^{m} \frac{2\,\theta_1 - 2\,z_{\mu,2}}{\hat{z}_{\mu,1} - \theta_2} + \alpha_\mu \left\{ \left(\hat{z}_{\mu,1}^u - \hat{z}_{\mu,1}\right)\left(\hat{z}_{\mu,1}^l - \hat{z}_{\mu,1}\right) \right. \\ & \left. + \left(\theta_1^u - \theta_1\right)\left(\theta_1^l - \theta_1\right) + \left(\theta_2^u - \theta_2\right)\left(\theta_2^l - \theta_2\right) \right\}.\end{aligned} \tag{19.36}$$

The explicit form of the α becomes :

$$\alpha_\mu = 2 \max_{\hat{z}_{\mu,1}\,\theta_1\,\theta_2} \frac{z_{\mu,2} - \theta_1 - \frac{\sqrt{2}}{2}\sqrt{\left(\hat{z}_{\mu,1} - \theta_2\right)^2 + 2\left(\theta_1 - z_{\mu,2}\right)^2}}{\left(\hat{z}_{\mu,1} - \theta_2\right)^3}. \tag{19.37}$$

Due to the complexity of the α term, it is not possible to determine where the maximum value occurs *a priori*. Therefore, within the αBB, a valid bound on α is determined using interval arithmetic methods (Adjiman and Floudas, 1996; Adjiman *et al.*, 1998a; Adjiman *et al.*, 1998b). Table 19.1 contains the statistics for each formulation.

In each attempted solution, bounds on the variables were set at: θ_1: [1:10], θ_2: [1-3.5:10] (depending on the size of the data set due to the discontinuity at $\theta_2 = \hat{z}_{\mu,1}$), and $\hat{z}_{\mu,i} : z_{\mu,i} \pm 0.05$. Bounds on the parameters were initially updated using the iterative method of step 3 in section 19.4, but bounds on the data set variables were held constant. Table 19.2 shows the results using the different formulations for one data set of 25 points. The results show that the α formulation converged quickly, while both the bilinear and unconstrained formulations did not achieve convergence in 1000 iterations. This result is quite counter-intuitive based on the problem formulation. First, an unconstrained formulation is much easier to solve from a local optimization standpoint, as is evident from the time per iteration. But in the framework of the αBB the problem is much more difficult due to the increased complexity of the nonconvexities. Analytical expressions for α were determined, but their form still required the use of approximations in calculating the numerical values. This resulted in

poor underestimation which is evident in the poor convergence rate. Secondly, the bilinear formulation possess an underestimator which represents the convex hull of each term in the constraint. This should lead to better convergence, but in generating this formulation, the number of variables in nonconvex terms was doubled and the number of constraints was greatly increased. Therefore twice as many variables may need to be branched on in order to achieve convergence. In generating the best possible formulation, not only should the forms of the nonconvex terms be considered, but also the number of terms and size of the variable set which participates in those terms.

Effect of the Size of the Data Set. The size of the data set has also a distinct effect on the convergence of the algorithm. Table 19.3 provides the obtained values for the parameters and objective function for each data set size. As the number of data points increases, the number of iterations required to solve the problem decreases, while the total solution time increases. The rate of increase in solution time is a result of the way the local solver, MINOS, scales with the number of variables. Figures 19.3 and 19.4 show this result graphically. Also noteworthy is how the resulting fitted parameters approach the true values as the number of data sets is increased. This illustrates well the unbiased estimates produced using maximum likelihood.

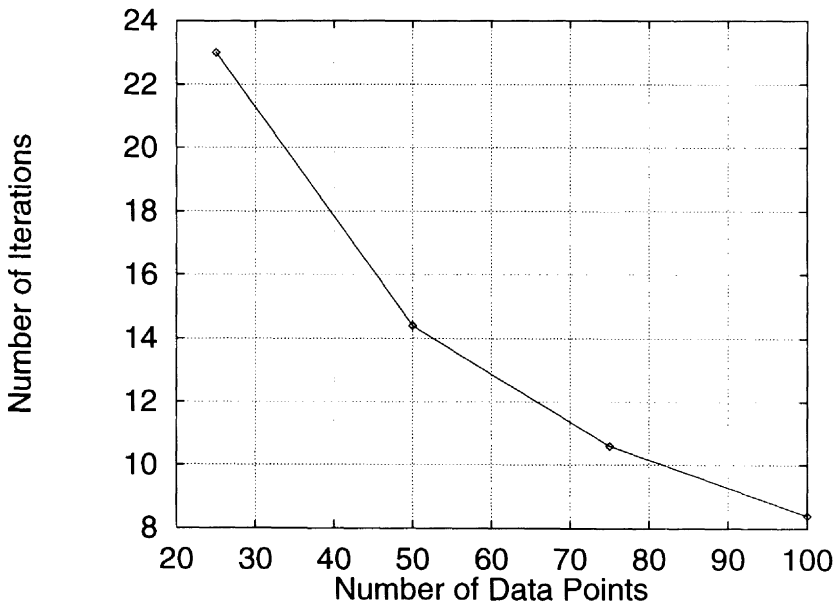


Figure 19.3. Number of iterations for Example 1 with different data set sizes. Results are the average using five separate, randomly generated data sets of each size.

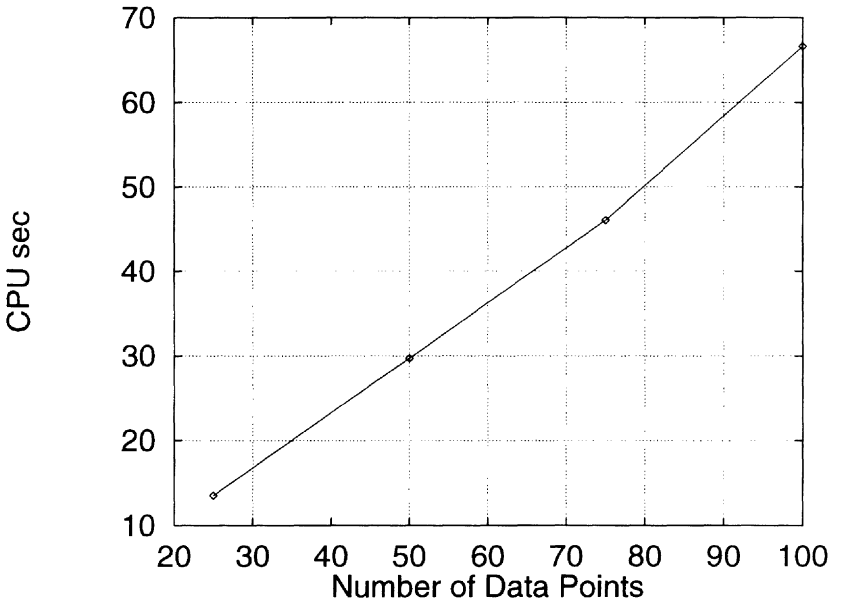


Figure 19.4. Computational time for Example 1 with different data set sizes. Results are the average using five separate, randomly generated data sets of each size.

19.5.2 Example 2: Kowalik Problem

This problem is taken from Moore *et al.* (1992). The mathematical model is:

$$z_1 = \theta_1 \frac{z_2^2 + z_2 \theta_2}{z_2^2 + z_2 \theta_3 + \theta_4}. \tag{19.38}$$

In this example, only z_1 contains error and therefore z_2 is treated as a constant. Two different formulations are possible.

Formulation 1. An unconstrained formulation becomes:

$$\min_{\theta} \sum_{\mu=1}^{11} \left[z_{\mu,1} - \frac{\theta_1 z_{\mu,2}^2 + \theta_1 \theta_2 z_{\mu,2}}{z_{\mu,2}^2 + z_{\mu,2} \theta_3 + \theta_4} \right]^2. \tag{19.39}$$

Each term in (19.39) will be treated as a term of general structure. The resulting formulation of the convex relaxation is:

$$\min_{\theta} \sum_{\mu=1}^{11} \left[z_{\mu,1} - \frac{\theta_1 z_{\mu,2}^2 + \theta_1 \theta_2 z_{\mu,2}}{z_{\mu,2}^2 + z_{\mu,2}\theta_3 + \theta_4} \right]^2 + \alpha_\mu \sum_{j=1}^{4} (\theta_j^u - \theta_j)(\theta_j^l - \theta_j). \tag{19.40}$$

The form of the nonconvex terms in (19.40) does not allow for the calculation of analytical values of α. Therefore in the solution of the lower bounding problem, numerical methods will be used to determine their values.

Formulation 2. A simpler constrained formulation can be generated by rewriting the model as:

$$\min_{\hat{z}_{\mu,1}\,\theta} \sum_{\mu=1}^{11} (\hat{z}_{\mu,1} - z_{\mu,1})^2 \tag{19.41}$$

s. t.

$$\hat{z}_{\mu,1} z_{\mu,2} - \theta_1 z_{\mu,2}^2 + z_{\mu,2}\hat{z}_{\mu,1}\theta_3 + \hat{z}_{\mu,1}\theta_4 - z_{\mu,2}\theta_2\theta_1 = 0.$$

In (19.41) all the nonconvexities are in the form of bilinear terms, making the determination of the convex underestimator much easier. The resulting lower bounding problem is:

$$\min_{\hat{z}_{\mu,1}\,\theta} \sum_{\mu=1}^{11} (\hat{z}_{\mu,1} - z_{\mu,1})^2 \tag{19.42}$$

s. t.

$$\begin{aligned} &\hat{z}_{\mu,1} z_{\mu,2} - \theta_1 z_{\mu,2}^2 + z_{\mu,2} w_{\mu,1} + w_{\mu,2} - z_{\mu,2} w_3 = 0 \\ &w_{\mu,1} \rightarrow \hat{z}_{\mu,1}\theta_3 \\ &w_{\mu,2} \rightarrow \hat{z}_{\mu,1}\theta_4 \\ &w_3 \rightarrow \theta_2\theta_1. \end{aligned}$$

Table 19.4 contains the statistics for both formulations.

The bounds on all the parameters are set at: [-0.2892, 0.2893] and on the data set variable, $\hat{z}_{\mu,2}$ of $z_{\mu,2} \pm 0.02$. Tables 19.5 and 19.6 show the local solutions found for the unconstrained and the constrained formulations, respectively. The unconstrained formulation, as in the previous example, did not converge to global optimality in 10,000 iterations. The constrained problem was solved using initial updating on both the parameters and the data set variables (initial bounds of $z_{\mu,2} \pm 0.01$ and $\delta_2 = 0.005$), as well as bounds updating at each iteration before branching on two randomly chosen candidates from the full set of variables. Convergence (absolute tolerance of 1e-8) was achieved in a total of 1905 αBB iterations in 319.5 CPU sec (two outer iterations).

Comparison with Interval Analysis Methods. Moore *et al.* (1992) employed an interval analysis approach and obtained a solution with an objective value enclosed by [0.00030748585, 0.00030748613] using an unconstrained formulation. Their solution required 4057.1 CPU sec. on a SPARC station SLC using 545,490 function evaluations, 492,923 Jacobian evaluations, and 200,627 Hessian evaluations.

19.5.3 Example 3: Respiratory Mechanical Model

This example is taken from Csendes and Ratz (1995). The model takes the form of:

$$z_\mu = \left(\theta_1 + \frac{\theta_2}{\omega_\mu^{\theta_3}}\right) + \jmath\left(\omega_\mu\,\theta_4 - \frac{\theta_5}{\omega_\mu^{\theta_3}}\right) \tag{19.43}$$

where $\jmath$ refers to $\sqrt{-1}$, and $\omega_\mu = \frac{\mu\,\pi}{20}$. In formulating this problem, the real and imaginary parts of $\hat{z}_\mu$ will be handled separately. Let $\hat{z}_\mu^r$ represent the real part and $\hat{z}_\mu^i$ the imaginary. For this problem two separate formulations are possible.

Formulation 1. An unconstrained formulation is written as:

$$\min_\theta \sum_{\mu=1}^{6} \left[\left(z_\mu^r - \theta_1 - \theta_2\,\omega_\mu^{-\theta_3}\right)^2 + \left(z_\mu^i - \omega_\mu\,\theta_4 + \theta_5\,\omega_\mu^{-\theta_3}\right)^2\right]. \tag{19.44}$$

Each of the above square terms are treated as general nonconvex terms and an α underestimator is used. The resulting convex relaxation is :

$$\begin{aligned} \min_\theta \quad & \sum_{\mu=1}^{6}\left(z_\mu^r - \theta_1 - \theta_2\,\omega_\mu^{-\theta_3}\right)^2 \\ + \quad & \sum_{\mu=1}^{6}\alpha_{\mu,1}\sum_{j=1}^{3}\left(\theta_j^u - \theta_j\right)\left(\theta_j^l - \theta_j\right) \\ + \quad & \sum_{\mu=1}^{6}\left(z_\mu^i - \omega_\mu\,\theta_4 + \theta_5\,\omega_\mu^{-\theta_3}\right)^2 \\ + \quad & \sum_{\mu=1}^{6}\alpha_{\mu,2}\sum_{j=3}^{5}\left(\theta_j^u - \theta_j\right)\left(\theta_j^l - \theta_j\right). \end{aligned} \tag{19.45}$$

Due to the nature of the nonconvexities in (19.45), analytical values of α were not easily identified. Therefore either numerical methods or fixed values need to be used in the solution.

Formulation 2. A second, constrained formulation can be generated with the use of the substitution $b_\mu = \omega_\mu^{-\theta_3}$. The resulting formulation is:

$$\min_{\hat{\mathbf{z}}_\mu\,\theta} \sum_{\mu=1}^{6} \left[\left(\hat{z}_\mu^r - z_\mu^r\right)^2 + \left(\hat{z}_\mu^i - z_\mu^i\right)^2 \right] \tag{19.46}$$

s. t.

$$\begin{aligned} \hat{z}_\mu^i - \omega_\mu\,\theta_4 + \theta_5\,b_\mu &= 0 \\ \hat{z}_\mu^r - \theta_1 - \theta_2\,b_\mu &= 0 \\ b_\mu - \omega_\mu^{-\theta_3} &\leq 0 \\ -b_\mu + \omega_\mu^{-\theta_3} &\leq 0. \end{aligned}$$

The nonconvexities are either in the form of bilinearities, or univariate concave terms ($-\omega_\mu^{-\theta_3}$ is concave, therefore the corresponding positive term is convex). The resulting formulation of the relaxed problem is:

$$\min_{\hat{\mathbf{z}}_\mu\,\theta} \sum_{\mu=1}^{6} \left[\left(\hat{z}_\mu^r - z_\mu^r\right)^2 + \left(\hat{z}_\mu^i - z_\mu^i\right)^2 \right] \tag{19.47}$$

s. t.

$$\begin{aligned} \hat{z}_\mu^i - \omega_\mu\,\theta_4 + w_{\mu,1} &= 0 \\ \hat{z}_\mu^r - \theta_1 - w_{\mu,2} &= 0 \\ b_\mu - \omega_\mu^{-\theta_3} &\leq 0 \\ -b_\mu + \left\{ \omega_\mu^{-\theta_3^l} + \frac{\omega_\mu^{-\theta_3^u} - \omega_\mu^{-\theta_3^l}}{\theta_3^u - \theta_3^l} \left(\theta_3 - \theta_3^l\right) \right\} &\leq 0 \\ w_{\mu,1} &\rightarrow \theta_5\,b_\mu \\ w_{\mu,2} &\rightarrow \theta_2\,b_\mu. \end{aligned}$$

Table 19.7 provides the problem statistics for each of the two different formulations.

The bounds used were [0.0, 1.0] for $\theta_{1,2,4,5}$ and [1.1, 1.3] for θ_3. Since the fitted data variables either do not appear in the problem (unconstrained formulation), or appear only in linear and convex terms (constrained formulation) their bounds do not have any effect on the underestimation. The bounds are therefore set at $z_{\mu,i} \pm (1 + \jmath)$ and no updating is performed.

Table 19.8 shows the results using both formulations. Using numerically calculated values for α, for the unconstrained formulation, convergence was not achieved in 10,000 iterations. The use of constant α values results in convergence to the known global solution, but theoretical guarantees do not hold since the convexity of the underestimator is not ensured. On the other hand, the constrained formulation converges quickly with theoretical guarantees. The global solution found (using the constrained formulation) was θ = [0.606298, 0.556761, 1.13181, 0.750199, 0.621899], with an objective value of 0.212460. This global solution was obtained in 1.38 CPU sec.

Comparison with Interval Analysis Methods. Csendes and Ratz (1995) obtained a solution with an objective value enclosed by [0.20788313, 0.21756222] using interval methods to solve the unconstrained formulation. Their solution required about 8 hours on a SUN SPARCSTATION with 222,275 objective function evaluations, and 618,406 partial derivative evaluations.

Summary

In this chapter we discussed a global optimization approach based on the αBB principles tailored to the *error-in-variables* parameter estimation problem for nonlinear algebraic models. Three distinct areas: variable bounds, branching method, and problem formulation, were outlined and their effect on the performance of the proposed algorithm was illustrated via three computational studies on problems taken from the literature. For additional computational studies the reader is directed to Esposito and Floudas (1998), while test problems in parameter estimation of algebraic and dynamic models can be found in the recent book by Floudas *et al.* (1999). Recently, Esposito and Floudas (1999) introduced a novel global optimization approach (i) for the parameter estimation and data reconcilliation of differential algebraic models, and (ii) for optimal control problems.

	Alpha		Bilinear		Unconstrained	
	Upper	Lower	Upper	Lower	Upper	Lower
Total Variables	2m+2	2m+2	2m+2	5m+3	2m+2	2m+2
Nonconvex Variables	m+1	-	2m+2	-	m+2	-
Nonconvex Terms	m	-	3m+1	-	m	-
Total Constraints	2m	2m	m	13m+4	-	-
Linear Constraints	0	0	0	13m+4	-	-

Table 19.1. Formulation Statistics for Example 1. In each case, m refers to the number of data points used in the estimation.

Formulation	Iterations	Rel. Convergence	CPU sec	sec/iter
Alpha	21	1%	11.2	0.54
Bilinear	1000	4.65%	540.2	0.54
Unconstrained	1000	>100%	282.8	0.28

Table 19.2. Convergence of different formulations of Example 1 using a 25 point data set.

Data Points	θ_1	θ_2	Obj. Value
25	2.022 ± 0.048	5.70 ± 1.38	$4.68 \pm 0.24 \times 10^{-3}$
50	2.010 ± 0.013	5.79 ± 0.27	$9.74 \pm 0.16 \times 10^{-3}$
75	2.009 ± 0.007	5.80 ± 0.15	$1.47 \pm 0.13 \times 10^{-2}$
100	1.999 ± 0.003	6.05 ± 0.07	$1.98 \pm 0.11 \times 10^{-2}$

Table 19.3. Parameter estimates for Example 1 with different sizes of data sets. Results presented are means and standard deviations using five separate, randomly generated data sets of each size.

	Constrained		Unconstrained	
	Upper	Lower	Upper	Lower
Total Variables	15	38	4	4
Nonconvex Variables	15	-	4	-
Nonconvex Terms	23	-	11	-
Total Constraints	11	103	-	-
Linear Constraints	0	103	-	-

Table 19.4. Formulation Statistics for Example 2.

Objective	θ_1	θ_2	θ_3	θ_4
0.03075×10^{-2}	0.19283	0.19088	0.12314	0.13578
0.12250×10^{-2}	0.22347	$-0.28920^{\dagger}$	0.03645	-0.10512
0.16262×10^{-2}	0.23050	-0.13221	0.25584	-0.05299
7.54451×10^{-2}	0.11559	$-0.28920^{\dagger}$	$-0.28920^{\dagger}$	$-0.28920^{\dagger}$

Table 19.5. Local solutions for Example 2, unconstrained formulation. (†) Variable is at its lower bound for the problem.

Objective	θ_1	θ_2	θ_3	θ_4
0.03075×10^{-2}	0.19283	0.19088	0.12314	0.13578
0.12250×10^{-2}	0.22347	$-0.28920^{\dagger}$	0.03645	-0.10512
0.20697×10^{-2} ‡	0.23329	-0.18019	0.15986	-0.06347
0.21918×10^{-2} ‡	0.23266	-0.13022	0.20372	-0.04461
0.22631×10^{-2} ‡	0.23353	-0.10108	0.23365	-0.03411
0.23198×10^{-2} ‡	0.23372	-0.07067	0.26351	-0.02338
0.24601×10^{-2} ‡	0.23375	-0.08323	0.25156	-0.02784

Table 19.6. Local solutions for Example 2, constrained formulation. (†) Variable is at its lower bound for the problem, (‡) one or more data set variables were at their bounds.

	Constrained		Unconstrained	
	Upper	Lower	Upper	Lower
Total Variables	23	35	5	5
Nonconvex Variables	9	-	5	-
Nonconvex Terms	18	-	12	-
Total Constraints	24	72	-	-
Linear Constraints	0	48	-	-

Table 19.7. Formulation statistics for Example 3.

Formulation	α values	Iterations	CPU sec
Unconstrained	Numerical	10000†	687.6
	1	908	51.1
	0.1	65	3.89
	0.01	9	0.661
Constrained	Analytical	11	1.38

Table 19.8. Convergence Rates for Example 3. An absolute tolerance of 0.01 is used for convergence. (†) Did not converge, absolute difference of 0.0640.

Data		Fitted Values
$z_{\mu,1}$	$1/z_{\mu,2}$	$\hat{z}_{\mu,1}$
0.1957	0.25	0.1944
0.1947	0.5	0.1928
0.1735	1	0.1824
0.1600	2	0.1489
0.0844	4	0.0928
0.0627	6	0.0624
0.0456	8	0.0456
0.0342	10	0.0355
0.0323	12	0.0288
0.0235	14	0.0241
0.0246	16	0.0207

Table 19.9. Data and fitted values for Example 2.

Data	Fitted Values
$5 - 5j$	$5.13 - 4.93j$
$3 - 2j$	$2.67 - 2.07j$
$2 - j$	$1.91 - 1.10j$
$1.5 - 0.5j$	$1.54 - 0.58j$
$1.2 - 0.2j$	$1.34 - 0.23j$
$1.1 - 0.1j$	$1.20 + 0.04j$

Table 19.10. Data and fitted values for Example 3.

IV Nonlinear and Mixed-Integer Optimization

20 INTRODUCTION TO NONLINEAR AND MIXED-INTEGER OPTIMIZATION

This chapter presents an introduction to the class of mixed-integer nonlinear MINLP optimization problems. Section 20.1 presents the motivation for studying MINLP problems. Section 20.2 discusses the mathematical formulation and outlines the theoretical and algorithmic challenges. Section 20.3 provides an overview of the local MINLP optimization algorithms which can address convex MINLP problems rigorously. Finally, section 20.4 outlines the global MINLP optimization approaches for nonconvex MINLP optimization problems.

20.1 MOTIVATION

A wide range of nonlinear optimization problems involve integer or discrete variables in addition to the continuous variables. These classes of optimization problems arise from a variety of applications and are denoted as Mixed–Integer Nonlinear Programming MINLP problems.

The integer variables can be used to model, for instance, sequences of events, alternative candidates, existence or non-existence of units (in their zero–one representation), while discrete variables can model, for instance, different equipment sizes. The continuous variables are used to model the input–output and interaction relationships among individual units/operations and different interconnected systems.

The nonlinear nature of these mixed–integer optimization problems may arise from (i) nonlinear relations in the integer domain exclusively (e.g., prod-

C. A. Floudas, *Deterministic Global Optimization*
© Springer Science+Business Media Dordrecht 2000

ucts of binary variables in the quadratic assignment model), (ii) nonlinear relations in the continuous domain only (e.g., complex nonlinear input–output model in a distillation column or reactor unit), (iii) nonlinear relations in the joint integer–continuous domain (e.g., products of continuous and binary variables in the scheduling/planning of batch processes and retrofit of heat recovery systems). The book by Nemhauser and Wolsey (1988) studies mixed–integer linear optimization and combinatorial optimization, while the book by Floudas (1995) studies mixed-integer nonlinear optimization problems.

The coupling of the integer domain and the continuous domain with their associated nonlinearities make the class of MINLP problems very challenging from the theoretical, algorithmic, and computational point of view. Mixed–Integer Nonlinear Optimization problems are encountered in a variety of applications in all branches of engineering and applied science, applied mathematics, and operations research. These represent very important and active research areas that include:

Process Synthesis

- Heat Exchanger Networks
- Retrofit of Heat Recovery Systems
- Distillation Sequencing
- Mass Exchange Networks
- Reactor–based Systems
- Reactor–Separator–Recycle Systems
- Utility Systems
- Total Process Systems
- Metabolic Engineering

Process Design

- Reactive Distillation
- Design of Dynamic Systems
- Plant Layout
- Environmental Design

Process Synthesis and Design Under Uncertainty

- Uncertainty Analysis
- Dynamic Systems
- Batch plant design

Molecular Design

- Solvent Selection

Design of Polymers and Refrigerants

Property Prediction Under Uncertainty

Interaction of Design, Synthesis and Control

Steady State Operation

Dynamic Operation

Process Operations

Scheduling of Multiproduct Plants

Design and Retrofit of Multiproduct Plants

Synthesis, Design and Scheduling of Multipurpose Plants

Planning Under Uncertainty

Facility Location and Allocation

Facility Planning and Scheduling

Topology of Transportation Networks

The applications in the area of *Process Synthesis* in Chemical Engineering include: (i) the synthesis of grassroot heat recovery networks (Floudas and Ciric, 1989; Ciric and Floudas, 1990; Ciric and Floudas, 1989; Yee *et al.*, 1990a; Yee and Grossmann, 1990; Yee *et al.*, 1990b); (ii) the retrofit of heat exchanger systems (Ciric and Floudas, 1990; Papalexandri and Pistikopoulos, 1993); (iii) the synthesis of distillation–based separation systems (Paules and Floudas, 1988; Viswanathan and Grossmann, 1990; Aggarwal and Floudas, 1990; Aggarwal and Floudas, 1992; Paules and Floudas, 1992; Novak *et al.*, 1996); (iv) the synthesis of mass exchange networks (Gupta and Manousiouthakis, 1993; Papalexandri *et al.*, 1994); (v) the synthesis of complex reactor networks (Kokossis and Floudas, 1990; Kokossis and Floudas, 1994a; Kokossis and Floudas, 1994a; Schweiger and Floudas, 1998c); (vi) the synthesis of reactor–separator–recycle systems (Kokossis and Floudas, 1991); (vii) the synthesis of utility systems (Kalitventzeff and Marechal, 1988); (viii) the synthesis of total process systems (Kocis and Grossmann, 1988; Kocis and Grossmann, 1989a; Kravanja and Grossmann, 1990; Daichendt and Grossmann, 1994a; Daichendt and Grossmann, 1994b; Papalexandri and Pistikopoulos, 1996; Kravanja and Grossmann, 1996); and (viiii) the analysis and synthesis of metabolic pathways (Dean and Dervakos, 1996; Hatzimanikatis *et al.*, 1996a; Hatzimanikatis *et al.*, 1996b; Petkov and Maranas, 1997a). Reviews of the mixed–integer nonlinear optimization frameworks and applications in Process Synthesis are provided in Grossmann (1990), Floudas (1995), Grossmann (1996b), and Adjiman *et al.* (1998c), while algorithmic advances for logic and global optimization in Process Synthesis are reviewed in Floudas and Grossmann (1995).

The MINLP applications in the area of *Process Design* include : (i) reactive distillation processes (Ciric and Gu, 1994); (ii) design of dynamic systems

(Barton *et al.*, 1998; Allgor and Barton, 1997; Schweiger and Floudas, 1997; Schweiger and Floudas, 1998b); (iii) plant layout systems (Penteado and Ciric, 1996; Georgiadis *et al.*, 1997); and (iv) environmentally benign systems (Ciric and Huchette, 1993; Stefanis *et al.*, 1997).

The MINLP applications in the area of *Process Synthesis and Design Under Uncertainty* include : (i) deterministic and stochastic uncertainty analysis (Grossmann and Floudas, 1987; Acevedo and Pistikopoulos, 1996; Dua and Pistikopoulos, 1998); (ii) design of dynamic systems under uncertainty (Dimitriadis and Pistikopoulos, 1995; Mohideen *et al.*, 1996); and (iii) design of batch processes under uncertainty (Pistikopoulos and Ierapetritou, 1995; Ierapetritou and Pistikopoulos, 1996; Harding and Floudas, 1997; Petkov and Maranas, 1998).

In the area of *Molecular Design*, the MINLP applications include : (i) the computer–aided molecular design aspects of selecting the best solvents (Odele and Macchietto, 1993); (ii) design of polymers and refrigerants (Maranas, 1996; Churi and Achenie, 1996; Churi and Achenie, 1997; Duvedi and Achenie, 1997; Raman and Maranas, 1999; Camarda and Maranas, 1999; Vaidyaraman and Maranas, 1999); and (iii) property prediction under uncertainty (Maranas, 1997).

The MINLP applications in the area of *Interaction of Synthesis, Design and Control* include : (i) studies under steady state operation of chemical processes (Luyben and Floudas, 1994a; Luyben and Floudas, 1994b; Papalexandri and Pistikopoulos, 1994a; Papalexandri and Pistikopoulos, 1994b); and (ii) studies under dynamic operation (Mohideen *et al.*, 1996; Mohideen *et al.*, 1997; Schweiger and Floudas, 1998b).

Applications of MINLP approaches have also emerged in the area of *Process Operations* and include: (i) short term scheduling of batch and semi-continuous processes (Zhang and Sargent, 1994; Mockus and Reklaitis, 1996), (ii) the design of multiproduct plants (Grossmann and Sargent, 1979; Birewar and Grossmann, 1989; Birewar and Grossmann, 1990); (iii) the synthesis, design and scheduling of multipurpose plants (Vaselenak *et al.*, 1987; Vaselenak *et al.*, 1987; Faqir and Karimi, 1990; Papageorgaki and Reklaitis, 1990; Papageorgaki and Reklaitis, 1990; Wellons and Reklaitis, 1991; Wellons and Reklaitis, 1991; Sahinidis and Grossmann, 1991; Fletcher *et al.*, 1991; Barbosa-Povoa and Macchietto, 1994; Xia and Macchietto, 1997); and (v) planning under uncertainty (Ierapetritou *et al.*, 1996; Ierapetritou and Pistikopoulos, 1994; Ierapetritou and Pistikopoulos, 1996; Liu and Sahinidis, 1997; Petkov and Maranas, 1997b). Reviews of the advances in the design, scheduling and planning of batch plants can be found in Reklaitis (1991), Grossmann *et al.* (1992), while a collection of recent contributions can be found in the proceedings of the 1998 FOCAPO meeting.

MINLP applications received significant attention in other engineering disciplines. These include (i) the facility location in a multiattribute space (Ganish *et al.*, 1983); (ii) the optimal unit allocation in an electric power system Bertsekas *et al.* (1983); (iii) the facility planning of an electric power genera-

tion (Bloom, 1983; Rouhani *et al.*, 1985); (iv) the chip layout and compaction (Dorneigh and Sahinidis, 1995); (v) the topology optimization of transportation networks (Hoang, 1982); and (vi) the optimal scheduling of thermal generating units (Geromel and Belloni, 1986).

20.2 MATHEMATICAL DESCRIPTION

The general algebraic MINLP formulation can be stated as:

$$\begin{aligned} \min_{\boldsymbol{x},\boldsymbol{y}} \quad & f(\boldsymbol{x},\boldsymbol{y}) \\ s.t. \quad & \boldsymbol{h}(\boldsymbol{x},\boldsymbol{y}) = \boldsymbol{0} \\ & \boldsymbol{g}(\boldsymbol{x},\boldsymbol{y}) \leq \boldsymbol{0} \\ & \boldsymbol{x} \in \boldsymbol{X} \subseteq \Re^n \\ & \boldsymbol{y} \in \boldsymbol{Y} \text{ integer.} \end{aligned} \tag{20.1}$$

Here $\boldsymbol{x}$ represents a vector of n continuous variables (e.g., flows, pressures, compositions, temperatures, sizes of units), and $\boldsymbol{y}$ is a vector of integer variables (e.g., alternative solvents or materials); $\boldsymbol{h}(\boldsymbol{x},\boldsymbol{y}) = \boldsymbol{0}$ denote the m equality constraints (e.g., mass, energy balances, equilibrium relationships); $\boldsymbol{g}(\boldsymbol{x},\boldsymbol{y}) \leq \boldsymbol{0}$ are the p inequality constraints (e.g., specifications on purity of distillation products, environmental regulations, feasibility constraints in heat recovery systems, logical constraints); and $f(\boldsymbol{x},\boldsymbol{y})$ is the objective function (e.g., annualized total cost, profit, thermodynamic criteria).

Remark 20.2.1 *The integer variables $\boldsymbol{y}$ with given lower and upper bounds*

$$\boldsymbol{y}^L \leq \boldsymbol{y} \leq \boldsymbol{y}^U$$

can be expressed through 0–1 variables (i.e., binary) denoted as, $\boldsymbol{z}$, by the following formula:

$$y = y^L + z_1 + 2z_2 + 4z_3 + \ldots + 2^{N-1}z_N$$

where N is the minimum number of 0–1 variables needed. This minimum number is given by:

$$N = 1 + INT\left\{\frac{\log\left(y^U - y^L\right)}{\log 2}\right\}$$

where the INT function truncates its real argument to an integer value.

Then, formulation (20.1) can be written in terms of 0–1 variables:

$$\begin{aligned} \min_{\boldsymbol{x},\boldsymbol{y}} \quad & f(\boldsymbol{x},\boldsymbol{y}) \\ \text{s.t.} \quad & \boldsymbol{h}(\boldsymbol{x},\boldsymbol{y}) = 0 \\ & \boldsymbol{g}(\boldsymbol{x},\boldsymbol{y}) \leq 0 \\ & \boldsymbol{x} \in \boldsymbol{X} \subseteq \Re^n \\ & \boldsymbol{y} \in \{0,1\}^q \end{aligned} \tag{20.2}$$

where $\boldsymbol{y}$ now is a vector of q 0–1 variables (e.g., existence of a process unit ($y_i = 1$) or non–existence ($y_i = 0$)).

20.2.1 Challenges in MINLP

Dealing with mixed–integer nonlinear optimization models of the form (20.1) or (20.2) present two major challenges. These difficulties are associated with the nature of the problem, namely, the combinatorial domain ($\boldsymbol{y}$–domain) and the continuous domain ($\boldsymbol{x}$–domain).

As the number of binary variables $\boldsymbol{y}$ in (20.2) increase, one is faced with a large combinatorial problem, and the complexity analysis results characterize MINLP problems as **NP**–complete (Nemhauser and Wolsey, 1988). At the same time, due to the nonlinearities the MINLP problems are in general nonconvex which implies the potential existence of multiple local solutions. The determination of a global solution of the nonconvex MINLP problems is also **NP**–hard, since even the global optimization of constrained nonlinear programming problems can be NP–hard (Pardalos and Schnitger, 1988), and even quadratic problems with one negative eigenvalue are NP–hard (Pardalos and Vavasis, 1991). An excellent book on complexity issues for nonlinear optimization is the one by Vavasis (1991).

Despite the aforementioned discouraging results from complexity analysis, which are worst–case results, significant progress has been achieved in the MINLP area from the theoretical, algorithmic, and computational perspective. As a result, several algorithms have been proposed for convex and nonconvex MINLP models, their convergence properties have been investigated, and a large number of applications now exist that cross the boundaries of several disciplines. In the sequel, we will discuss these developments.

20.3 OVERVIEW OF LOCAL OPTIMIZATION APPROACHES FOR CONVEX MINLP MODELS

A representative collection of local MINLP algorithms developed for solving convex MINLP models of the form (20.1) or restricted classes of (20.2) includes the following:

1. Generalized Benders Decomposition, GBD, (Geoffrion, 1972; Paules and Floudas, 1989; Floudas *et al.*, 1989);

2. Outer Approximation, OA, (Duran and Grossmann, 1986a);

3. Outer Approximation with Equality Relaxation, OA/ER, (Kocis and Grossmann, 1987);

4. Outer Approximation with Equality Relaxation and Augmented Penalty, OA/ER/AP, (Viswanathan and Grossmann, 1990);

5. Generalized Outer Approximation, GOA, (Fletcher and Leyffer, 1994);

6. Generalized Cross Decomposition, GCD, (Holmberg, 1990);

7. Branch and Bound, BB, (Beale, 1977; Gupta, 1980; Ostrovsky *et al.*, 1990; Borchers and Mitchell, 1991; Quesada and Grossmann, 1992; Fletcher and Leyffer, 1998);

8. Feasibility Approach, FA, (Mawengkang and Murtagh, 1986);

9. Extended Cutting Plane, ECP, (Westerlund *et al.*, 1994; Westerlund and Pettersson, 1995);

10. Logic-Based Approaches, (Turkay and Grossmann, 1996; Vecchietti and Grossmann, 1999).

In the pioneering work of Geoffrion (1972) on the Generalized Benders Decomposition, GBD, two sequences of updated upper (nonincreasing) and lower (nondecreasing) bounds are created that converge within ϵ in a finite number of iterations. The upper bounds correspond to solving subproblems in the $\boldsymbol{x}$ variables by fixing the $\boldsymbol{y}$ variables, while the lower bounds are based on duality theory.

The Outer Approximation, OA, addresses problems with nonlinear inequalities, and creates sequences of upper and lower bounds as the GBD, but it has the distinct feature of using primal information, that is the solution of the upper bound problems, so as to linearize the objective and constraints around that point. The lower bounds in OA are based upon the accumulation of the linearized objective function and constraints, around the generated primal solution points.

The OA/ER algorithm, extends the OA to handle nonlinear equality constraints by relaxing them into inequalities according to the sign of their associated multipliers.

The OA/ER/AP algorithm introduces an augmented penalty function in the lower bound subproblems of the OA/ER approach.

The Generalized Outer Approximation, GOA, extends the OA to the MINLP problems that the GBD addresses and introduces exact penalty functions.

The Generalized Cross Decomposition, GCD, simultaneously utilizes primal and dual information by exploiting the advantages of Dantzig–Wolfe and Generalized Benders decomposition.

An overview of these local MINLP algorithms and extensive theoretical, algorithmic, and applications of GBD, OA, OA/ER, OA/ER/AP, GOA, and GCD algorithms can be found in the book by Floudas (1995).

The branch and bound, BB, approaches start by solving the continuous relaxation of the MINLP and subsequently perform an implicit enumeration where a subset of the 0–1 variables is fixed at each node. The lower bound corresponds to the NLP solution at each node and it is used to expand on the node with the lowest lower bound or it is used to eliminate nodes if the lower bound exceeds the current upper bound. If the continuous relaxation, NLP in most cases with the exception of the algorithm of Quesada and Grossmann (1992) where an LP problem is obtained, of the MINLP has a 0–1 solution for the y variables, then the BB algorithm will terminate at that node. With a similar argument, if a tight NLP relaxation results in the first node of the tree, then the number of nodes that would need to be eliminated can be low. However, loose NLP relaxations may result in having a large number of NLP subproblems to be solved. The algorithm terminates when the lowest lower bound is within a pre-specified tolerance of the best upper bound.

The feasibility approach, FA, rounds the relaxed NLP solution to an integer solution with the least local degradation by successively forcing the superbasic variables to become nonbasic based on the reduced cost information. The premise of this approach is that the problems to be treated are sufficiently large so that techniques requiring the solution of several NLP relaxations, such as the branch-and-bound approach, have prohibitively large costs. They therefore wish to account for the presence of the integer variables in the formulation and solve the mixed-integer problem directly. This is achieved by fixing most of the integer variables to one of their bounds (the *nonbasic* variables) and allowing the remaining small subset (the *basic* variables) to take discrete values in order to identify feasible solutions. After each iteration, the reduced costs of the variables in the nonbasic set are computed to measure their effect on the objective function. If a change causes the objective function to decrease, the appropriate variables are removed from the nonbasic set and allowed to vary for the next iteration. When no more improvement in the objective function is possible, the algorithm is terminated. This strategy leads to the identification of a local solution.

The cutting plane algorithm proposed by Kelley (1960) for NLP problems has been extended to MINLPs (Westerlund *et al.*, 1994; Westerlund and Pettersson, 1995). The ECP algorithm relies on the linearization of one of the nonlinear constraints at each iteration and the solution of the increasingly tight MILP made up of these linearizations. The solution of the MILP problem provides a new point on which to base the choice of the constraint to be linearized for the next iteration of the algorithm. The ECP does not require the solution of any NLP problems for the generation of an upper bound. As a result, a large number of linearizations are required for the approximation of highly nonlinear problems and the algorithm does not perform well in such cases. Due to the use of linearizations, convergence to the global optimum solution is guaranteed only for problems involving inequality constraints which are convex in the x and relaxed y-space.

An alternative to the direct solution of the MINLP problem was proposed by Turkay and Grossmann (1996). Their approach stems from the work of Kocis and Grossmann (1989) on a modeling/decomposition strategy which avoids the zero-flows generated by the non-existence of a unit in a process network. The first stage of the algorithm is the reformulation of the MINLP into a generalized disjunctive program. A vector of Boolean variables indicate the status of a disjunction (True or False) and are associated with the alternatives. The set of disjunctions allows the representation of several alternatives. A set of logical relationships between the Boolean variables is introduced. Instead of resorting to binary variables within a single model, the disjunctions are used to generate a different model for each alternative. Since all continuous variables associated with the non-existing alternatives are set to zero, this representation helps to reduce the size of the problems to be solved. Two algorithms are suggested by Turkay and Grossmann (1996). They are logic-based variants of the Outer Approximation and Generalized Benders Decomposition. Vecchietti and Grossmann (1999) introduced LOGMIP, a computer code for disjunctive programming and MINLP problems, and studied modeling alternatives and process synthesis applications.

20.4 OVERVIEW OF GLOBAL OPTIMIZATION APPROACHES FOR NONCONVEX MINLP MODELS

In the previous section we discussed *local* MINLP algorithms which are applicable to convex MINLP models. While identification of the global solution for convex problems can be guaranteed, a local solution is often obtained for nonconvex problems. A number of global MINLP algorithms that have been developed to address different types of nonconvex MINLPs are presented in this section. These include :

1. Branch-and-Reduce Approach, (Ryoo and Sahinidis, 1995);
2. Interval Analysis based Approach, (Vaidyanathan and El-Halwagi, 1996);
3. Extended Cutting Plane Approach, (Westerlund *et al.*, 1994; Westerlund *et al.*, 1998);
4. Reformulation/Spatial Branch-and-Bound Approach, (Smith and Pantelides, 1996; Smith and Pantelides, 1999);
5. Hybrid Branch-and-Bound and Outer Approximation Approach, (Zamora and Grossmann, 1998a; Zamora and Grossmann, 1998b);
6. The SMIN-αBB Approach, (Adjiman *et al.*, 1997c; Adjiman *et al.*, 1998d);
7. The GMIN-αBB Approach, (Adjiman *et al.*, 1997c; Adjiman *et al.*, 1998d).

In the sequel, we will briefly discuss the approaches 1-7 while the SMIN-αBB and the GMIN-αBB approaches will be discussed in more detail in Chapters 21 and 22, respectively.

20.4.1 Branch-and-Reduce algorithm

Ryoo and Sahinidis (1995) extended the scope of branch-and-bound algorithms to problems for which valid convex underestimating NLPs can be constructed for the nonconvex relaxations. The range of application of the proposed algorithm encompasses bilinear problems and separable problems involving functions for which convex underestimators can be built (McCormick, 1976; Al-Khayyal, 1990). Because the nonconvex NLPs must be underestimated at each node, convergence can only be achieved if the continuous variables are branched on. A number of tests are suggested to accelerate the reduction of the solution space. They are summarized in the following.

Optimality Based Range Reduction Tests. For the first set of tests, an upper bound U on the nonconvex MINLP must be computed and a convex lower bounding NLP must be solved to obtain a lower bound L. If a bound constraint for variable x_i, with $x_i^L \leq x_i \leq x_i^U$, is active at the solution of the convex NLP and has multiplier $\lambda_i^* > 0$, the bounds on x_i can be updated as follows:

1. If $x_i - x_i^U = 0$ at the solution of the convex NLP and $\kappa_i = x_i^U - \frac{U-L}{\lambda_i^*}$ is such that $\kappa_i > x_i^L$, then $x_i^L = \kappa_i$.
2. If $x_i - x_i^L = 0$ at the solution of the convex NLP and $\kappa_i = x_i^L + \frac{U-L}{\lambda_i^*}$ is such that $\kappa_i < x_i^U$, then $x_i^U = \kappa_i$.

If neither bound constraint is active at the solution of the convex NLP for some variable x_j, the problem can be solved by setting $x_j = x_j^U$ or $x_j = x_j^L$. Tests similar to those presented above are then used to update the bounds on x_j.

Feasibility Based Range Reduction Tests. In addition to ensuring that tight bounds are available for the variables, the underestimators of the constraints are used to generate new constraints for the problem. Consider the constraint $g_i(\boldsymbol{x}, \boldsymbol{y}) \leq 0$. If its underestimating function $\underline{g}_i(\boldsymbol{x}, \boldsymbol{y}) = 0$ at the solution of the convex NLP and its multiplier is $\mu_i^* > 0$, the constraint

$$\underline{g}_i(\boldsymbol{x}, \boldsymbol{y}) \geq -\frac{U-L}{\mu_i^*}$$

can be included in subsequent problems.

The branch-and-reduce algorithm has been tested on a set of small problems.

20.4.2 Interval Analysis Based Approach

An approach based on interval analysis was proposed by Vaidyanathan and El-Halwagi (1996) to solve to global optimality problems with a twice-differentiable objective function and once-differentiable constraints. Interval arithmetic allows the computation of guaranteed ranges for these functions (Moore, 1979;

Ratschek and Rokne, 1984; Neumaier, 1990). The approach relies on the same concepts of successive partitioning of the domain and bounding of the objective function, while the branching takes place on the discrete and continuous variables. The main difference with the branch-and-bound algorithms is that bounds on the problem solution in a given domain are not obtained through optimization. Instead, they are based on the range of the objective function in the domain under consideration, as computed with interval arithmetic. As a consequence, these bounds may be quite loose and efficient fathoming techniques are required in order to enhance convergence. Vaidyanathan and El-Halwagi (1996) suggested node fathoming tests and branching strategies which are outlined in the sequel. Convergence is declared when best upper and lower bounds are within a pre-specified tolerance and when the width of the corresponding region is below a pre-specified tolerance.

Node Fathoming Tests. The *upper-bound test* is a classical criterion used in all branch-and-bound schemes: if the lower bound for a node is greater than the best upper bound for the MINLP, the node can be fathomed.

The *infeasibility test* is also used by all branch-and-bound algorithms. However, the identification of infeasibility using interval arithmetic differs from its identification using optimization schemes. An inequality constraint $g_i(\boldsymbol{x}, \boldsymbol{y}) \leq 0$ is declared infeasible if its interval inclusion over the current domain, is positive. If a constraint is found to be infeasible, the current node is fathomed.

The *monotonicity test* is used in interval-based approaches. If a region is feasible, the monotonicity properties of the objective function can be tested. For this purpose, the inclusions of the gradients of the objective with respect to each variable are evaluated. If all the gradients have a constant sign for the current region, the objective function is monotonic and only one point needs to be retained from the current node.

The *nonconvexity test* is used to test the existence of a solution (local or global) within a region. If such a point exists, the Hessian matrix of the objective function at this point must be positive semi-definite. A sufficient condition is the non-negativity of at least one of the diagonal elements of its interval Hessian matrix.

Vaidyanathan and El-Halwagi (1996) suggested two additional tests to accelerate the fathoming process. The first is denoted as *lower bound test.* It requires the computation of a valid lower bound on the objective function through a method other than interval arithmetic. If the upper bound at a node is less than this lower bound, the region can be eliminated. The second test, the *distrust-region method,* aims to help the algorithm identify infeasible regions so that they can be removed from consideration. Based on the knowledge of an infeasible point, interval arithmetic is used to identify an infeasible hypercube centered on that point.

Branching Strategies. The variable with the widest range is selected for branching. It can be a continuous or a discrete variable. In order to determine where to split the chosen variable, a relaxation of the MINLP is solved locally.

Case 1 - Continuous Branching Variable : If the optimal value of the continuous branching variable, x^*, is equal to one of the variable bounds, branch at the midpoint of the interval. Otherwise, branch at $x^* - \beta$, where β is a very small scalar.

Case 2 - Discrete Branching Variable : If the optimal value of the discrete branching variable, y^*, is equal to the upper bound on the variable, define a region with $y = y^*$ and one with $y^L \leq y \leq y^* - 1$, where y^L is the lower bound on y. Otherwise, create two regions $y^L \leq y \leq int(y^*)$ and $int(y^*)+1 \leq y \leq y^U$, where y^U is the upper bound on y.

This algorithm has been tested on a small example problem and a molecular design problem (Vaidyanathan and El-Halwagi, 1996).

20.4.3 Extended Cutting Plane for Pseudoconvex MINLPs

The use of the ECP algorithm for nonconvex MINLP problems was suggested in Westerlund *et al.* (1994), using a modified algorithmic procedure as described in Westerlund *et al.* (1998). The main changes occur in the generation of new constraints for the MILP at each iteration (Step 4). In addition to the construction of the linear function $l_k(\boldsymbol{x}, \boldsymbol{y})$ at iteration k, the following steps are taken:

1. Remove all constraints for which $l_i(\boldsymbol{x}^k, \boldsymbol{y}^k) > g_{j_i}(\boldsymbol{x}^k, \boldsymbol{y}^k)$. These correspond to linearizations which did not underestimate the corresponding nonlinear constraint at all points due to the presence of nonconvexities.

2. Replace all constraints for which $l_i(\boldsymbol{x}^k, \boldsymbol{y}^k) = g_{j_i}(\boldsymbol{x}^k, \boldsymbol{y}^k) = 0$ by their linearization around $(\boldsymbol{x}^k, \boldsymbol{y}^k)$.

3. If constraint i is such that $g_i(\boldsymbol{x}^k, \boldsymbol{y}^k) > 0$, add its linearization around $(\boldsymbol{x}^k, \boldsymbol{y}^k)$.

The convergence criterion is also modified. In addition to the test used in Step 3, the following two conditions must be met:

1. $(\boldsymbol{x}^k - \boldsymbol{x}^{k-1})^T(\boldsymbol{x}^k - \boldsymbol{x}^{k-1}) \leq \delta$, a pre-specified tolerance.

2. $\boldsymbol{y}^k - \boldsymbol{y}^{k-1} = 0$.

The ECP algorithm for pseudoconvex MINLPs has been used to address a trim loss problem arising in the paper industry (Westerlund *et al.*, 1998). A comparative study between the Outer Approximation, the Generalized Benders Decomposition and the Extended Cutting Plane algorithm for convex MINLPs was presented in Skrifvars *et al.* (1996).

20.4.4 Reformulation/Spatial Branch-and-Bound Algorithm

A global optimization algorithm of the branch-and-bound type was proposed in Smith and Pantelides (1996), (1997, 1999). It can be applied to problems in which the objective and constraints are functions involving any combination of binary arithmetic operations (addition, subtraction, multiplication and division) and functions that are either concave over the entire solution space (such as ln) or convex over this domain (such as exp).

The algorithm starts with an automatic reformulation of the original nonlinear problem into a problem that involves only linear, bilinear, linear fractional, simple exponentiation, univariate concave and univariate convex terms. This is achieved through the introduction of new constraints and variables. The reformulated problem is then solved to global optimality using a branch-and-bound approach. Its special structure allows the construction of a convex relaxation at each node of the tree. It should be noted that due to the introduction of many new constraints and variables the size of the convex relaxation of the reformulated problem increases substantially even for modest size problems. The integer variables can be handled in two ways during the generation of the convex lower bounding problem. The integrality condition on the variables can be relaxed to yield a convex NLP which can then be solved globally. Alternatively, the integer variables can be treated directly and the convex lower bounding MINLP can be solved using a branch-and-bound algorithm. This second approach is more computationally intensive but is likely to result in tighter lower bounds on the global optimum solution.

In order to obtain an upper bound on the optimum solution, a local MINLP algorithm can be used. Alternatively, the MINLP can be transformed to an equivalent nonconvex NLP by relaxing the integer variables. For example, a variable $y \in \{0, 1\}$ can be replaced by a continuous variable $z \in [0, 1]$ by including the constraint $z - z \cdot z = 0$.

This algorithm has been applied to reactor selection, distillation column design, nuclear waste blending, heat exchanger network design and multilevel pump configuration problems.

20.4.5 Hybrid Branch-and-Bound and Outer Approximation

Zamora and Grossmann (1998a) proposed a global optimization MINLP approach for the synthesis of heat exchanger networks without stream splitting. This approach is a hybrid branch-and-bound with outer approximation. It is based on two alternative convex underestimators for the heat transfer area. The first type of these convex underestimators along with the variable bounds and techniques for the bound contraction are based on a thermodynamic analysis. The second type is based on a relaxation and transformation so as to employ specific underestimation schemes. These convex underestimators result in a convex MINLP that is solved using the Outer Approximation approach and which provides valid lower bounds on the global solution. This approach has been applied to five heat exchanger network examples that employ the

MINLP model of Yee and Grossmann (1990) that contains linear constraints and nonconvex objective function.

Zamora and Grossmann (1998b) introduced a deterministic branch and contract approach for structured process systems that have univariate concave, bilinear and linear fractional terms. They proposed properties of the contraction operation and studied their effect on several applications.

20.4.6 The SMIN-αBB Algorithm

The SMIN-αBB global optimization algorithm, proposed by Adjiman *et al.* (1997c), (1998d), is designed to solve to global optimality mathematical models where the binary/integer variables appear linearly and hence separably from the continuous variables and/or appear in at most bilinear terms, while nonlinear terms in the continuous variables appear separably from the binary/integer variables. These mathematical models take the form :

$$\begin{aligned} \min_{\boldsymbol{x},\boldsymbol{y}} \quad & f(\boldsymbol{x}) + \boldsymbol{x}^T A_0 \boldsymbol{y} + \boldsymbol{c}_0^T \boldsymbol{y} \\ \text{s.t.} \quad & \boldsymbol{h}(\boldsymbol{x}) + \boldsymbol{x}^T A_1 \boldsymbol{y} + \boldsymbol{c}_1^T \boldsymbol{y} = 0 \\ & \boldsymbol{g}(\boldsymbol{x}) + \boldsymbol{x}^T A_2 \boldsymbol{y} + \boldsymbol{c}_2^T \boldsymbol{y} \leq 0 \\ & \boldsymbol{x} \in \boldsymbol{X} \subseteq \Re^n \\ & \boldsymbol{y} \in \boldsymbol{Y} \text{ integer} \end{aligned}$$

where $\boldsymbol{c}_0^T$, $\boldsymbol{c}_1^T$ and $\boldsymbol{c}_2^T$ are constant vectors, A_0, A_1 and A_2 are constant matrices and $f(\boldsymbol{x})$, $\boldsymbol{h}(\boldsymbol{x})$ and $\boldsymbol{g}(\boldsymbol{x})$ are functions with continuous second-order derivatives.

We will discuss the theoretical, algorithmic and computational studies of the SMIN-αBB algorithm in detail in Chapter 21.

20.4.7 The GMIN-αBB Algorithm

The GMIN-αBB global optimization algorithm proposed by Adjiman *et al.* (1997c), (1998d), operates within a branch-and-bound framework. The main difference with the algorithms of Gupta and Ravindran (1985), Ostrovsky *et al.* (1990) and Borchers and Mitchell (1991) is its ability to identify the global optimum solution of a much larger class of problems of the form

$$\begin{aligned} \min_{\boldsymbol{x},\boldsymbol{y}} \quad & f(\boldsymbol{x},\boldsymbol{y}) \\ \text{s.t.} \quad & \boldsymbol{h}(\boldsymbol{x},\boldsymbol{y}) = \boldsymbol{0} \\ & \boldsymbol{g}(\boldsymbol{x},\boldsymbol{y}) \leq \boldsymbol{0} \\ & \boldsymbol{x} \in \boldsymbol{X} \subseteq \Re^n \\ & \boldsymbol{y} \in \mathcal{N}^q \end{aligned}$$

where $\mathcal{N}$ is the set of non-negative integers and the only condition imposed on the functions $f(\boldsymbol{x}, \boldsymbol{y})$, $\boldsymbol{g}(\boldsymbol{x}, \boldsymbol{y})$ and $\boldsymbol{h}(\boldsymbol{x}, \boldsymbol{y})$ is that their continuous relaxations possess continuous second-order derivatives. This increased applicability results from the use of the αBB global optimization algorithm for continuous twice-differentiable NLPs (Androulakis *et al.*, 1995; Adjiman and Floudas, 1996; Adjiman *et al.*, 1998a; Adjiman *et al.*, 1998b).

We will discuss the theoretical, algorithmic and computational studies of the GMIN-αBB Algorithm in detail in Chapter 22.

Summary

In this chapter, we discussed the need for studying MINLP problems that are widespread in many application areas. We presented the mathematical formulation of MINLP models and we reviewed the main local and global optimization approaches for the solution of the convex and nonconvex MINLP models, respectively.

IV Nonlinear and Mixed-Integer Optimization

21 THE SMIN-αBB APPROACH : THEORY AND COMPUTATIONS

This chapter presents the Special structure Mixed Integer Nonlinear αBB global optimization approach, denoted as SMIN-αBB, for nonconvex MINLP problems. Section 21.1 describes the classes of mathematical problems to which the SMIN-αBB is applicable. Section 21.2 focuses on the generation of valid lower bounds. Section 21.3 discusses the generation of upper bounds. Section 21.4 presents the strategies for the selection of the branching variables. Section 21.5 discusses the updates of the variable bounds via an optimization-based and an interval analysis based approach. Section 21.6 outlines the algorithmic steps of the SMIN-αBB approach and provides an illustration. Finally, in section 21.7 computational studies are presented on (i) small literature problems, and (ii) heat exchanger network synthesis problems. The material in this chapter is based on the work of Adjiman *et al.* (1997c), (1998d).

21.1 MATHEMATICAL FORMULATION

The Special structure Mixed Integer Nonlinear αBB, SMIN-αBB, algorithm guarantees ϵ-convergence to a global solution of MINLPs belonging to the class:

C. A. Floudas, *Deterministic Global Optimization*
© Springer Science+Business Media Dordrecht 2000

$$
\begin{aligned}
\min_{\boldsymbol{x},\boldsymbol{y}} \quad & f(\boldsymbol{x}) + \boldsymbol{x}^T A_f \boldsymbol{y} + c_f^T \boldsymbol{y} \\
s.t. \quad & g_i(\boldsymbol{x}) + \boldsymbol{x}^T A_{g,i} \boldsymbol{y} + c_{g,i}^T \boldsymbol{y} \leq 0, \; i = 1, \ldots, m \\
& h_i(\boldsymbol{x}) + \boldsymbol{x}^T A_{h,i} \boldsymbol{y} + c_{h,i}^T \boldsymbol{y} = 0, \; i = 1, \ldots, p \\
& \boldsymbol{x} \in [\boldsymbol{x}^L, \boldsymbol{x}^U], \quad \boldsymbol{y} \in \{0,1\}^q
\end{aligned}
\tag{21.1}
$$

where $f(\boldsymbol{x})$, $g_i(\boldsymbol{x})$, $i = 1, \ldots, m$, and $h_i(\boldsymbol{x}), i = 1, \ldots, p$ are twice continuously differentiable functions, m is the number of inequality constraints, p is the number of equality constraints, q is the dimension of the binary variable vector, A_f, $A_{g,i}$ and $A_{h,i}$ are matrices of size $n \times q$, and c_f, $c_{g,i}$ and $c_{h,i}$ are vectors of size q.

As we discussed in Chapter 20 and in the book by Floudas (1995), there exist many engineering and applied science problems that can be expressed in this form. The SMIN-αBB global optimization approach belongs to the class of deterministic global optimization methods in which valid lower and upper bounds on the global solution are generated at each iteration. The sequence of the lower bounds is monotonically nondecreasing while the sequence of updated upper bounds is monotonically nonincreasing. The improved lower bounds correspond to refinement of the domain into smaller domains in which convex underestimators are generated. This chapter presents the main steps of the SMIN-αBB branch-and-bound algorithm.

21.2 GENERATION OF VALID LOWER BOUNDS

To generate a valid lower bound, a relaxed problem which can be solved to global optimality is formulated from problem (21.1). Such a relaxed problem is derived by constructing convex underestimators for the nonconvex functions $f(\boldsymbol{x})$, $g_i(\boldsymbol{x})$ and $h_i(\boldsymbol{x})$. The rigorous convexification/relaxation strategy used in the αBB algorithm for nonconvex continuous problems (Adjiman and Floudas, 1996; Adjiman *et al.*, 1998a) allows the formulation of the desired lower bounding MINLP. As we discussed in Chapter 12, this scheme is based on a decomposition of the functions into a sum of terms with special mathematical structure, such as linear, convex, bilinear, trilinear, fractional, fractional trilinear, univariate concave and general nonconvex terms. A different convex relaxation technique is applied for each class of term:

(i) the linear and convex terms are unchanged,

(ii) the bilinear terms are underestimated by their convex envelope, which consists of four linear inequalities for each bilinear term (see Chapters 2 and 12),

(iii) the trilinear, fractional and fractional trilinear terms are replaced by a new variable subject to a set of convex inequalities (see Chapters 2 and 12),

(iv) the univariate concave terms are underestimated by their convex envelope consisting of a linear function going through the bounds (see Chapters 2 and 12), and

(v) a general nonconvex term $F_{NC}(\boldsymbol{x})$ can be underestimated by the αBB type of convex underestimators (see Chapters 11 and 12) given by :

$$F_{NC}(\boldsymbol{x}) - \sum_{i=1}^{n} \alpha_i (x_i - x_i^L)(x_i^U - x_i), \tag{21.2}$$

where the α_i's are positive scalars such that $H_{F_{NC}}(\boldsymbol{x}) + 2\text{diag}(\alpha_i)$ is positive semi-definite for all $\boldsymbol{x} \in [\boldsymbol{x}^L, \boldsymbol{x}^U]$, where $H_{F_{NC}}(\boldsymbol{x})$ is the Hessian matrix of the general nonconvex term.

As we discussed in Chapter 12, the α_i parameters can be rigorously obtained by deriving an interval Hessian matrix $[H_{F_{NC}}]$ such that $H_{F_{NC}}(\boldsymbol{x}) \in [H_{F_{NC}}]\ \forall \boldsymbol{x} \in [\boldsymbol{x}^L, \boldsymbol{x}^U]$. This allows the relaxation of the convexity condition to $[H_{F_{NC}}] + 2\text{diag}(\alpha_i)$ positive semi-definite. Appropriate α_i's can then be derived using one of the techniques presented in Chapter 12 (see also Adjiman *et al.* (1996), Adjiman and Floudas (1996) and Adjiman *et al.* (1998a)).

To formulate the lower bounding problem, the nonconvex terms in the objective function and the inequality constraints are simply replaced by their convex underestimators. This leads to the generation of a problem with a lower convex objective function and an enlarged convex feasible region. The treatment of the equality constraints is based on the following two cases :

Case I : Equality constraints that contain univariate concave terms, convex terms, and general nonconvex terms. These are assigned to a set P_N and each constraint $h_i(\boldsymbol{x}) + \boldsymbol{x}^T A_{h,i} \boldsymbol{y} + c_{h,i}^T \boldsymbol{y} = 0$, $i \in P_N$, is reformulated as two inequality constraints,

$$\begin{aligned} h_i(\boldsymbol{x}) + \boldsymbol{x}^T A_{h,i} \boldsymbol{y} + c_{h,i}^T \boldsymbol{y} &\leq 0 \\ h_i^-(\boldsymbol{x}) - \boldsymbol{x}^T A_{h,i} \boldsymbol{y} - c_{h,i}^T \boldsymbol{y} &\leq 0 \end{aligned} \tag{21.3}$$

where $h_i^-(\boldsymbol{x})$ denotes $-h_i(\boldsymbol{x})$.

Case II : Equality constraints that involve only bilinear, trilinear, fractional and fractional trilinear terms. These can be underestimated directly since such terms are replaced by a new variable whose bounds are given by a set of convex inequality constraints which define an enlarged feasible region. The resulting equalities, together with the linear equalities, are assigned to a set P_L.

Based on this analysis, a general formulation for the convex lower bounding problem is given by

$$
\begin{aligned}
\min_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{w}} \quad & \breve{f}(\boldsymbol{x},\boldsymbol{w}) + \boldsymbol{x}^T A_f \boldsymbol{y} + c_f^T \boldsymbol{y} \\
& \breve{g}_i(\boldsymbol{x},\boldsymbol{w}) + \boldsymbol{x}^T A_{g,i} \boldsymbol{y} + c_{g,i}^T \boldsymbol{y} \leq 0, \; i = 1, \ldots, m \\
& \breve{h}_i(\boldsymbol{x},\boldsymbol{w}) + \boldsymbol{x}^T A_{h,i} \boldsymbol{y} + c_{h,i}^T \boldsymbol{y} = 0, \; i \in P_L \\
& \breve{h}_i(\boldsymbol{x},\boldsymbol{w}) + \boldsymbol{x}^T A_{h,i} \boldsymbol{y} + c_{h,i}^T \boldsymbol{y} \leq 0, \; i \in P_N \\
& \breve{\bar{h}}_i^{-}(\boldsymbol{x},\boldsymbol{w}) - \boldsymbol{x}^T A_{h,i} \boldsymbol{y} - c_{h,i}^T \boldsymbol{y} \leq 0, \; i \in P_N \\
& \mathcal{C}(\boldsymbol{x},\boldsymbol{w}) \leq 0 \\
& \boldsymbol{x} \in [\boldsymbol{x}^L, \boldsymbol{x}^U], \; \boldsymbol{y} \in \{\boldsymbol{y}^L, \boldsymbol{y}^U\}, \; \boldsymbol{w} \in [\boldsymbol{w}^L, \boldsymbol{w}^U]
\end{aligned}
\tag{21.4}
$$

where the $\breve{}$ superscript denotes the convex underestimator of the specified function valid for the current domain $[\boldsymbol{x}^L, \boldsymbol{x}^U]$; $\boldsymbol{w}$ denotes the set of variables that replace bilinear, trilinear, fractional and fractional trilinear terms in the continuous variables; and $\mathcal{C}(\boldsymbol{x},\boldsymbol{w})$ denotes the set of additional linear constraints that arise from the underestimation of these terms.

Remark 21.2.1 Note that this lower bounding problem, after the transformation of the bilinear terms between the continuous and binary variables, corresponds to a convex MINLP problem that can be solved with local MINLP algorithms such as the Generalized Benders Decomposition, the Outer Approximation, and the Generalized Cross Decomposition which are part of MINOPT (Schweiger and Floudas, 1997).

21.3 GENERATION OF UPPER BOUNDS

Valid upper bounds may be obtained in several ways. For instance, a local solution of the nonconvex MINLP (21.1) constitutes a valid upper bound on the global optimum solution of that problem. The Generalized Benders Decomposition (GBD) (Benders, 1962; Geoffrion, 1972; Floudas *et al.*, 1989) or a standard MINLP branch-and-bound algorithm (Beale, 1977; Gupta and Ravindran, 1985; Ostrovsky *et al.*, 1990; Borchers and Mitchell, 1991; Quesada and Grossmann, 1992) may be used to obtain such a solution. When there are no mixed-bilinear terms, the outer-approximation with equality relaxation (OA/ER) (Duran and Grossmann, 1986a; Kocis and Grossmann, 1987) may also be used. Alternatively, the binary variables may be fixed to a combination of 0 and 1 values and the resulting nonconvex NLP may be solved locally.

Remark 21.3.1 Notice that in the αBB algorithm the upper bound is obtained by solving a nonconvex NLP locally, while in the SMIN-αBB algorithm, the local solution of a nonconvex NLP or MINLP is involved.

21.4 SELECTION OF BRANCHING VARIABLES

As in the αBB algorithm, a list of lower bounds for the regions that have not yet been partitioned is maintained. The region with the lowest lower bound is selected for further exploration at each iteration. Several strategies can be used to select the variable to be branched on at the chosen node, which are described in the sequel.

21.4.1 Continuous Variable Branching

This branching variable selection scheme exploits the direct relationship between the range of the continuous variables and the quality of the lower bounds by partitioning only on these variables, and employing the criteria of the least-reduced axis, the term measure or the variable measure.

In the *least-reduced axis rule*, the variable that has been least branched on has the greatest ratio

$$\frac{x_i^U - x_i^L}{x_{i,orig}^U - x_{i,orig}^L}, \tag{21.5}$$

where $x_{i,orig}^U$ and $x_{i,orig}^L$ are respectively the upper and lower bounds on variable x_i at the first node of the branch-and-bound and x_i^U and x_i^L are respectively the upper and lower bounds on variable x_i at the current node.

The *term measure μ_t criterion*, that reflects the tightness of the convex underestimator, can be computed for each nonconvex term and one of the variables participating in the term with the greatest measure can be selected for branching. The term measure can be taken as the separation distance between the nonconvex term and its underestimator at the solution of the convex lower bounding MINLP at the current node, or as the maximum separation distance between the nonconvex term and its underestimator.

The *measure of each variable's overall contribution* to the quality of the underestimators may be used to select a continuous branching variable. In this case, the variable measure μ_v^i for variable x_i is the summation of the term measures for all terms in which x_i participates. The variable with the largest measure is then selected for branching.

21.4.2 Sequential Branching: Binary-Continuous

This branching variable selection approach addresses the combinatorial aspects of the problem and its nonconvexity sequentially. In the first levels of the branch-and-bound tree, branching takes place on the binary variables only. This policy is followed until all binary variables have been fixed. Binary variable branching can go on for at most q levels of the branch-and-bound tree, where q is the number of binary variables. In instances where variable bound updates allow us to discard some of the values of the binary variables, fewer levels are needed to fix all binary variables.

The choice of a specific binary variable for branching is based on random selection or/and on a priority ranking of the variables. The highest ranking binary variables are those that influence the bounds on the greatest number of variables. In general, variables that play a similar role in determining the structure of the problem are assigned the same priority. The branching variable is chosen randomly from the highest ranking group.

An important aspect of sequential branching from the computational standpoint is that, once all the binary variables have been fixed, the problems that must be solved to obtain upper and lower bounds on the solution are *continuous* nonconvex and convex problems. The bounding of the nodes during the continuous variable branching phase is therefore less computationally intensive than during the binary branching phase.

21.4.3 Hybrid Branching

This branching variable selection approach also involves branching on the continuous and binary variables although the type is no longer determined from the level in the tree. To increase the impact of binary variable branching on the quality of the lower bound, such a variable is selected when a continuous relaxation of the problem indicates that the two newly created children nodes may have significantly different lower bounds. Thus, if one of the binary variables is close to 0 or 1 at a local solution of the continuous relaxation, it is branched on. The degree of closeness is an arbitrary parameter which is typically set to 0.1 or 0.2. If no "almost-integer" binary variable is found, a continuous variable is selected for branching using one of the rules described in Section 21.4.1.

21.5 UPDATES OF VARIABLE BOUNDS

Two methods have been developed to tighten the variable bounds of the continuous and binary variables. The first is an optimization-based approach in which a convex MINLP is formulated and solved for each bound to be updated. The second is an iterative approach based on interval arithmetic. This latter method can be modified to enable the tightening of binary variable bounds.

21.5.1 Optimization-based Approach

For every continuous variable x_i whose lower or upper bound is to be updated, the following optimization problem is formulated:

$$
\begin{aligned}
\min_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{w}} \;/\; \max_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{w}} \quad & x_i \\
s.t. \quad & \breve{f}(\boldsymbol{x},\boldsymbol{w}) + \boldsymbol{x}^T A_f \boldsymbol{y} + c_f^T \boldsymbol{y} \leq \overline{f}^* \\
& \breve{g}_i(\boldsymbol{x},\boldsymbol{w}) + \boldsymbol{x}^T A_{g,i} \boldsymbol{y} + c_{g,i}^T \boldsymbol{y} \leq 0, \; i = 1, \ldots, m \\
& \breve{h}_i(\boldsymbol{x},\boldsymbol{w}) + \boldsymbol{x}^T A_{h,i} \boldsymbol{y} + c_{h,i}^T \boldsymbol{y} = 0, \; i \in P_L \\
& \breve{h}_i(\boldsymbol{x},\boldsymbol{w}) + \boldsymbol{x}^T A_{h,i} \boldsymbol{y} + c_{h,i}^T \boldsymbol{y} \leq 0, \; i \in P_N \\
& \breve{h}_i^-(\boldsymbol{x},\boldsymbol{w}) - \boldsymbol{x}^T A_{h,i} \boldsymbol{y} - c_{h,i}^T \boldsymbol{y} \leq 0, \; i \in P_N \\
& \mathcal{C}(\boldsymbol{x},\boldsymbol{w}) \leq \boldsymbol{0} \\
& \boldsymbol{x} \in [\boldsymbol{x}^L, \boldsymbol{x}^U], \boldsymbol{y} \in \{\boldsymbol{y}^L, \boldsymbol{y}^U\}, \boldsymbol{w} \in [\boldsymbol{w}^L, \boldsymbol{w}^U]
\end{aligned}
\tag{21.6}
$$

where the $\breve{}$ superscript denotes the convex underestimator of the specified function valid for the current domain $[\boldsymbol{x}^L, \boldsymbol{x}^U]$; $\overline{f}^*$ denotes the best upper bound value on the global optimum solution; P_L denotes the set of equality constraints which involve only linear, bilinear, trilinear, fractional and fractional trilinear terms; P_N denotes the other equality constraints; $h_i^-(\boldsymbol{x})$ denotes $-h_i(\boldsymbol{x})$; $\boldsymbol{w}$ denotes the set of variables that replace bilinear, trilinear, fractional and fractional trilinear terms; and $\mathcal{C}(\boldsymbol{x},\boldsymbol{w})$ denotes the set of additional constraints that arise from the underestimation of these terms.

Remark 21.5.1 Notice that in addition to the set of convexified constraints, problem (21.6) includes the constraint

$$\breve{f}(\boldsymbol{x},\boldsymbol{w}) + \boldsymbol{x}^T A_f \boldsymbol{y} + c_f^T \boldsymbol{y} \leq \overline{f}^*, \tag{21.7}$$

based on the convexified objective function and the best known upper bound on the global optimum solution, $\overline{f}^*$. In order to get even tighter bounds, the latest improved bound replaces the old one in the bound constraint $\boldsymbol{x} \in [\boldsymbol{x}^L, \boldsymbol{x}^U]$ before the next bound update is performed. To fully reflect the reduction in range, the underestimators which depend on the improved bound should also be updated.

21.5.2 Interval-based Approach

Interval arithmetic (Moore, 1979; Neumaier, 1990; Hansen, 1992) can be used to update the binary and continuous variable bounds. This approach is based

on the interval evaluation of the constraints in the nonconvex MINLP as well as Eq. (21.7), the bound improvement constraint. The binary variables that have not yet been fixed are treated as continuous variables. The interval evaluation gives enclosures for the values of all constraints in the relaxed space $(\boldsymbol{x}, \boldsymbol{y}) \in [\boldsymbol{x}^L, \boldsymbol{x}^U] \times [\boldsymbol{y}^L, \boldsymbol{y}^U]$. Once a range of values $[F^L, F^U]$ is obtained for a function $F(\boldsymbol{x}, \boldsymbol{y})$ such that $F(\boldsymbol{x}, \boldsymbol{y}) \in [F^L, F^U] \ \ \forall (\boldsymbol{x}, \boldsymbol{y}) \in [\boldsymbol{x}^L, \boldsymbol{x}^U] \times [\boldsymbol{y}^L, \boldsymbol{y}^U]$, the feasibility of the constraint can be assessed. A necessary feasibility condition for the inequality constraint $F(\boldsymbol{x}, \boldsymbol{y}) \leq 0$ is that $F^L \leq 0$. Similarly, a necessary feasibility condition for the equality constraint $F(\boldsymbol{x}, \boldsymbol{y}) = 0$ is $0 \in [F^L, F^U]$. If the interval evaluation of a single constraint does not satisfy the appropriate necessary condition, the current region is infeasible. At a given node, the feasible region for the bound update problem is defined as :

$$\begin{aligned}
\mathcal{F} = \{(\boldsymbol{x}, \boldsymbol{y}) : f(\boldsymbol{x}) + \boldsymbol{x}^T A_f \boldsymbol{y} + c_f^T \boldsymbol{y} &\leq \overline{f}^*; \\
g_i(\boldsymbol{x}) + \boldsymbol{x}^T A_{g,i} \boldsymbol{y} + c_{g,i}^T \boldsymbol{y} &\leq 0, \ i = 1, \ldots, m; \\
h_i(\boldsymbol{x}) + \boldsymbol{x}^T A_{h,i} \boldsymbol{y} + c_{h,i}^T \boldsymbol{y} &= 0, \ i = 1, \ldots, p; \\
(\boldsymbol{x}, \boldsymbol{y}) \in [\boldsymbol{x}^L, \boldsymbol{x}^U] &\times [\boldsymbol{y}^L, \boldsymbol{y}^U]\}.
\end{aligned}$$

A bisection procedure is used to improve the variable bounds: for a lower (upper) bound, the aim is to identify an infeasible region on the lower (upper) end of the variable range. Initially, half the variable range is considered. The interval feasibility of the left (right) half is tested. If this region is infeasible, it is discarded and the right (left) half is then tested. Otherwise, the region is further split in two and the leftmost (rightmost) range is once again tested. This procedure is repeated until the original range has been determined to be entirely infeasible or until the range being tested is smaller than a user-specified tolerance. This approach is illustrated in Figure 21.1 for the lower bound of a variable originally in the range $[0, 4]$, with a tolerance of 1.

For a binary variable, the interval-based procedure for continuous variables can be modified to account for the nature of the variable. In this case, no iterations are needed: to update both bounds on binary variable y_i, the interval feasibility of $y_i = 0$ and $y_i = 1$ is tested. If both tests reveal infeasibility, the entire region is discarded; if one test only is feasible, y_i is fixed to the bound corresponding to the feasible test; and if both tests are feasible, the bounds on y_i remain unchanged.

Remark 21.5.2 The interval-based bound updates are less computationally intensive than the optimization-based strategy. However, the use of interval arithmetic often results in a larger relaxation of the constraints than the convexification/relaxation scheme used in formulating problem (21.6). Thus, at early levels of the branch-and-bound tree, when variable bound updates are most critical, the overestimation of the interval computations is most significant. As a result, the optimization-based approach leads to tighter bounds.

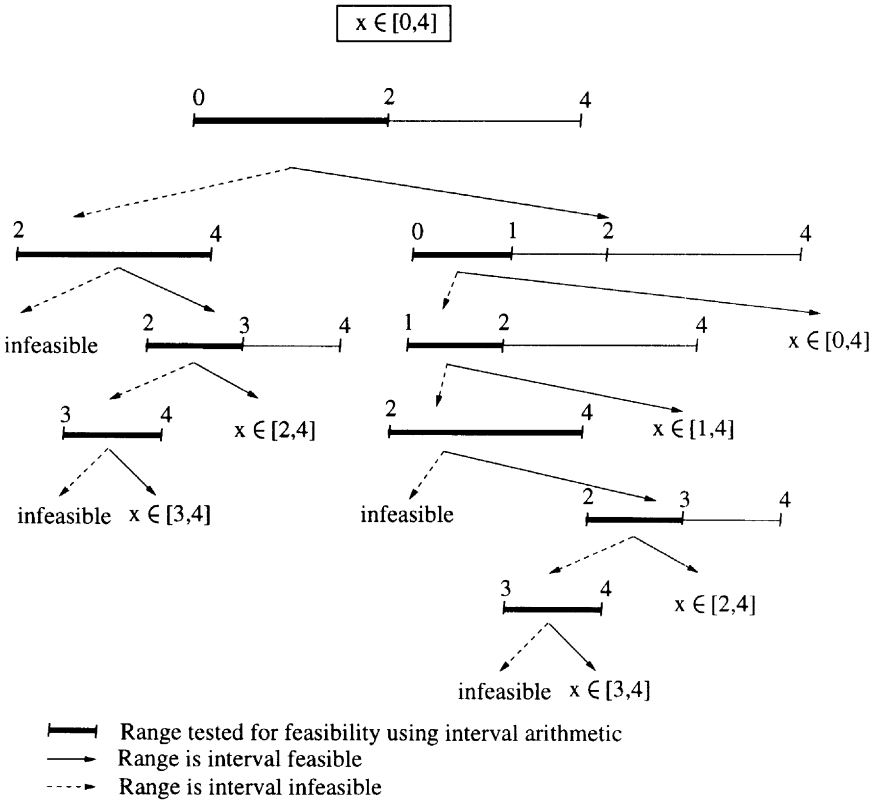


Figure 21.1. Interval bound update procedure for the lower bound on a variable in $[0, 4]$. The minimum range width for testing is 1.

21.6 THE SMIN-αBB ALGORITHM

In the SMIN-αBB algorithm, the upper and lower bounding procedures, and the branching and variable bound update strategies are embedded within a branch-and-bound algorithm. The flowchart shown in Figure 21.2 highlights the main features of the algorithm.

21.6.1 Illustrative Example

In order to illustrate the algorithmic procedure, a small example proposed by Kocis and Grossmann (1989) and recently studied by Adjiman *et al.* (1998d) is used. It is a simple design problem where one of two reactors must be chosen to produce a given product at the lowest possible cost. It involves two binary variables, one for each reactor, and seven continuous variables. In the following formulation, the constraint $z_1 + z_2 = 10$ has been used to tighten the bounds

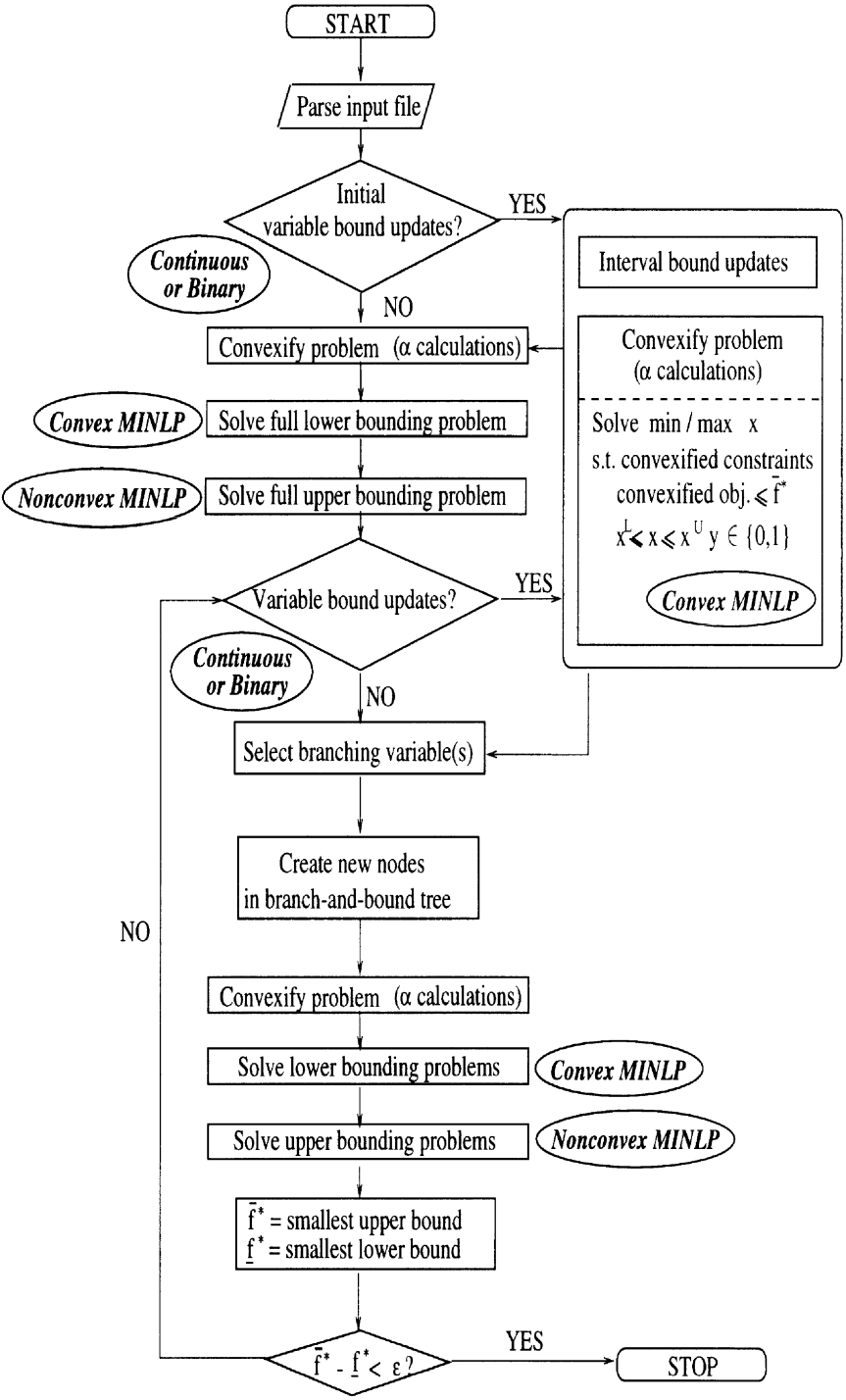


Figure 21.2. Flowchart for the SMIN-αBB algorithm

on z_1 and z_2:

$$\begin{array}{ll} \min & 7.5y_1 + 5.5y_2 + 7v_1 + 6v_2 + 5x \\ s.t. & z_1 - 0.9\left(1 - e^{-0.5v_1}\right)x_1 = 0 \\ & z_2 - 0.8\left(1 - e^{-0.5v_2}\right)x_2 = 0 \\ & x_1 + x_2 - x = 0 \\ & z_1 + z_2 = 10 \\ & v_1 - 10y_1 \leq 0 \\ & v_2 - 10y_2 \leq 0 \\ & x_1 - 12y_1 \leq 0 \\ & x_2 - 12y_2 \leq 0 \\ & y_1 + y_2 = 1 \\ & 0 \leq x_1, x_2 \leq 20; 0 \leq z_1, z_2 \leq 10 \\ & 0 \leq v_1, v_2 \leq 10; 0 \leq x \leq 20 \\ & (y_1, y_2) \in \{0,1\}^2. \end{array}$$

Because of the linear participation of the binary variables, the SMIN-αBB algorithm is well-suited to solve this nonconvex MINLP. The selection of a branching variable is random for the binary variables and is based on the term measures for the continuous variables. Bound updates are performed at every iteration and branching takes place on the binary variables first. The global solution of 99.2 is identified after eight iterations. At the global solution, the binary variable values are $y_1 = 1$ and $y_2 = 0$. The steps of the algorithm are shown in the branch-and-bound tree of Figure 21.3. The boldface numbers next to the nodes indicate the order in which the nodes were explored. The lower bound obtained by solving a convex relaxation of the nonconvex problem is indicated inside each node, and the branching variable selected for the node is also specified. The domain to which this branching variable is restricted is displayed along each branch. Note that the bounds shown are based on the results of the bound tightening procedure performed at each iteration. A black node indicates the lower bounding problem was found infeasible and a shaded node is fathomed because its lower bound is greater than the current upper bound on the solution.

At the first node, the initial lower bound is 11.4 and an upper bound of 99.2 is found. The binary variable y_1 is selected as a branching variable. The region $y_1 = 0$ is infeasible and can therefore be fathomed (black node), while an improved lower bound is found for $y_1 = 1$. This latter region is therefore chosen for exploration at the second iteration. Variable bound updates reveal that $y_2 = 1$ is infeasible so that y_2 can be fixed to zero. Branching on the continuous variables may now begin. The first selected variable is x_1 and regions $0 \leq x_1 \leq 10$ and $10 \leq x_1 \leq 20$ are created. Since the left region has the lowest lower bound (36.3), it is examined at iteration 3 by branching on x_1. Both children nodes are found to be infeasible. The algorithm proceeds to node 4 for which v_1 is selected as a branching variable. The right region, $5.6 \leq v_1 \leq 10$, is fathomed since it has a lower bound greater than 99.2. The algorithm progresses along the branch-and-bound tree until, at iteration 8, only

one node is left with a lower bound of 99.2. This is within the accuracy required for this run so the procedure is terminated.

21.7 COMPUTATIONAL STUDIES

In this section, we discuss several computational studies of the SMIN-αBB for (i) small literature problems and (ii) three heat exchanger network synthesis problems. For the solution of convex MINLPs, the SMIN-αBB algorithm relies on the implementations of the Generalized Benders Decomposition and the Outer Approximation which are part of MINOPT (Schweiger and Floudas, 1997).

21.7.1 Small Literature Examples

The following examples have been studied in the literature (Kocis and Grossmann, 1988; Floudas *et al.*, 1989; Ryoo and Sahinidis, 1995; Cardoso *et al.*, 1997) and all runs are performed on an HP-C160.

Example 1. This example, taken from Kocis and Grossmann (1988), involves three binary variables and two continuous variables. The formulation is given by :

$$\begin{aligned}
\min_{\boldsymbol{x},\boldsymbol{y}} \quad & 2x_1 + 3x_2 + 1.5y_1 + 2y_2 - 0.5y_3 \\
s.t. \quad & x_1^2 + y_1 = 1.25 \\
& x_2^{1.5} + 1.5y_2 = 3 \\
& x_1 + y_1 \leq 1.6 \\
& 1.333x_2 + y_2 \leq 3 \\
& -y_1 - y_2 + y_3 \leq 0 \\
& x_1, x_2, x_3 \geq 0 \\
& (y_1, y_2, y_3) \in \{0,1\}^3.
\end{aligned}$$

The value of the objective function at the global solution is 7.67 and the optimum solution vectors are $\boldsymbol{x}^* = (1.12, 1.31)^T$ and $\boldsymbol{y}^* = (0,1,1)^T$. The problem is solved by the SMIN-αBB after four iterations and 0.4 CPU seconds when using sequential branching, optimization-based bound updates for the continuous variables and interval-based bound updates for the binary variables.

Example 2. This example is example 6.6.5 from Floudas (1995). The formulation involves two continuous variables and only one binary variable.

$$\begin{aligned}
\min_{\boldsymbol{x},\boldsymbol{y}} \quad & -0.7y + 5(x_1 - 0.5)^2 + 0.8 \\
s.t. \quad & -e^{(x_1 - 0.2)} - x_2 \leq 0 \\
& x_2 + 1.1y \leq 1 \\
& x_1 - 1.2y \leq 0.2 \\
& 0.2 \leq x_1 \leq 1 \\
& -2.22554 \leq x_2 \leq -1 \\
& y \in \{0,1\}.
\end{aligned}$$

The global solution has an objective value of 1.07654 and the corresponding solution vectors are $\boldsymbol{x}^* = (0.94194, -2.1)^T$ and $y^* = 1$. Using the same strategy as in Example 1, the SMIN-αBB algorithm converges in nine iterations and 0.5 CPU seconds.

Example 3. This example was proposed by Yuan *et al.* (1988) and involves three continuous variables and four binary variables. The formulation is :

$$
\begin{aligned}
\min_{\boldsymbol{x},\boldsymbol{y}} \quad & (y_1-1)^2+(y_2-2)^2+(y_3-1)^2-\ln(y_4+1) \\
& \qquad +(x_1-1)^2+(x_2-2)^2+(x_3-3)^2 \\
s.t. \quad & y_1+y_2+y_3+x_1+x_2+x_3 \le 5 \\
& y_3^2+x_1^2+x_2^2+x_3^2 \le 5.5 \\
& y_1+x_1 \le 1.2 \\
& y_2+x_2 \le 1.8 \\
& y_3+x_3 \le 2.5 \\
& y_4+x_1 \le 1.2 \\
& y_2^2+x_2^2 \le 1.64 \\
& y_3^2+x_3^2 \le 4.25 \\
& y_2^2+x_3^2 \le 4.64 \\
& x_1,x_2,x_3 \ge 0 \\
& \boldsymbol{y} \in \{0,1\}^4.
\end{aligned}
$$

Since the binary variables participate nonlinearly in this problem, it must be reformulated to be solved by the SMIN-αBB algorithm. The new formulation involves seven continuous variables and four binary variables as well as four new linear equality constraints. This larger problem is a convex MINLP which can be solved using the OA/ER. The global solution has an objective value of 4.5796 and solution vectors $\boldsymbol{x}^* = (0.2, 0.8, 1.908)^T$ and $\boldsymbol{y}^* = (1,1,0,1)^T$. As expected, the SMIN-$\alpha$BB algorithm converges in only one iteration and 0.3 CPU seconds.

Example 4. This example was first presented by Berman and Ashrafi (1993) and involves three continuous variables and eight binary variables that participate in a highly nonlinear manner.

$$
\begin{aligned}
\min_{\boldsymbol{x},\boldsymbol{y}} \quad & -x_1x_2x_3 \\
s.t. \quad & x_1+0.1^{y_1}0.2^{y_2}0.15^{y_3}=1 \\
& x_2+0.05^{y_4}0.2^{y_5}0.15^{y_6}=1 \\
& x_3+0.02^{y_7}0.06^{y_8}=1 \\
& -y_1-y_2-y_3 \le -1 \\
& -y_4-y_5-y_6 \le -1 \\
& -y_7-y_8 \le -1 \\
& 3y_1+y_2+2y_3+3y_4+2y_5+y_6+3y_7+2y_8 \le 10 \\
& 0 \le x_1,x_2,x_3 \le 1 \\
& \boldsymbol{y} \in \{0,1\}^8.
\end{aligned}
$$

The global optimum solution has an objective function value of -0.94347 and solution vectors $\boldsymbol{x}^* = (0.97, 0.9925, 0.98)^T$ and $\boldsymbol{y}^* = (0,1,1,1,0,1,1,0)^T$. For the SMIN-$\alpha$BB algorithm to solve this problem, it must be reformulated through a logarithmic transformation. For instance, the first constraint is equivalent to the linear constraint

$$\ln(1-x_1) - y_1 \ \ln(0.1) - y_2 \ \ln(0.2) - y_3 \ \ln(0.15) = 0.$$

The SMIN-αBB algorithm solves the reformulated problem in eight iterations and 1.3 CPU seconds.

Example 5. This example is a purely integer nonconvex problem presented in Pörn *et al.* (1997) and involves two variables.

$$\begin{aligned} \min_{\boldsymbol{y}} \quad & 7y_1 + 10y_2 \\ s.t. \quad & y_1^{1.2} y_2^{1.7} - 7y_1 - 9y_2 \leq 24 \\ & -y_1 - 2y_2 \leq 5 \\ & -3y_1 + y_2 \leq 1 \\ & 4y_1 - 3y_2 \leq 11 \\ & y_1, y_2 \in [1,5] \cap \mathbb{N}. \end{aligned}$$

The global optimum solution is 31, with $(y_1, y_2) = (3,1)^T$. To be solved with the SMIN-αBB algorithm, this problem must be reformulated by expressing each integer variable y_i in terms of binary variables and new constraints:

$$\begin{aligned} & y_i = 1 + z_{i1} + 2z_{i2} + 4z_{i3} \\ & z_{i1} + z_{i3} \leq 1 \\ & z_{i1} + z_{i2} \leq 1 \\ & z_{i1}, z_{i2}, z_{i3} \in \{0,1\}. \end{aligned}$$

The y_i variables can then be treated as continuous variables and the nonconvex inequality can be underestimated using the α parameters.

The binary reformulation is solved in two iterations and 0.5 CPU seconds using the SMIN-αBB algorithm.

Example 6. This example, taken from Pörn *et al.* (1997), involves one continuous and one integer variable.

$$\begin{aligned} \min_{x,y} \quad & 3y - 5x \\ s.t. \quad & 2y^2 - 2y^{0.5} - 2x^{0.5}y^2 + 11y + 8x \leq 39 \\ & -y + x \leq 3 \\ & 2y + 3x \leq 24 \\ & 1 \leq x \leq 10 \\ & y \in [1,6] \cap \mathbb{N}. \end{aligned}$$

The reformulation in terms of binary variables is solved in four iterations and 0.5 CPU seconds by the SMIN-αBB algorithm. The optimal solution is -17, with $x^* = 4$ and $y^* = 1$.

21.7.2 *Heat exchanger network synthesis*

The performance of the SMIN-αBB algorithm is studied on three heat exchanger network synthesis problems, using the Chen approximation (see Floudas (1995)) of the logarithmic mean temperature difference or the arithmetic mean temperature difference in order to compute the area.

21.7.2.1 Example 1 : Use of the Chen approximation. This example studies the synthesis of a heat exchanger network involving two hot streams, two cold streams, one hot and one cold utility. The superstructure and formulation of Yee and Grossmann (1991) is used. The superstructure for this problem is shown in Figure 21.4 and it has two stages. There are 12 possible matches and therefore 12 binary variables. The annualized cost of the network is expressed as the summation of the utility costs, the fixed charges for the required heat-exchangers and an area-based cost for each heat-exchanger. The Chen approximation for the logarithmic mean of the temperature difference is used (Chen, 1987). The binary variables, which represent the existence of a given heat-exchanger, participate linearly in the problem. Also, all the constraints are linear.

Given the set ST of temperature locations, the number of stages NS, the set HP of hot process streams and the set CP of cold process streams, the general problem formulation is as follows.

$$
\begin{aligned}
\min \sum_{i\in HP} C_{CU}Q_{CU,i} + \sum_{j\in CP} C_{HU}Q_{HU,j} \\
+ \sum_{i\in HP}\sum_{j\in CP}\sum_{k\in ST} CF_{ij}z_{ijk} + \sum_{i\in HP} CF_{i,CU}z_{CU,i} + \sum_{j\in CP} CF_{j,HU}z_{HU,j} \\
+ \sum_{i\in HP}\sum_{j\in CP}\sum_{k\in ST} \frac{C_{ij}Q_{ijk}}{U_{ij}[\Delta T_{ijk}\Delta T_{ijk+1}(\Delta T_{ijk}+\Delta T_{ijk+1})/2]^{\frac{1}{3}}} \\
+ \sum_{i\in HP} \frac{CA,i,CUQ_{CU,i}}{U_{CU,i}[\Delta T_{CU,i}(T_{out,i}-T_{in,CU})(\Delta T_{CU,i}+T_{out,i}-T_{in,CU})/2]^{\frac{1}{3}}} \\
+ \sum_{j\in CP} \frac{CA,j,HUQ_{HU,j}}{U_{HU,j}[\Delta T_{HU,j}(T_{in,HU}-T_{out,j})(\Delta T_{HU,j}+T_{in,HU}-T_{out,j})/2]^{\frac{1}{3}}}
\end{aligned}
$$

$$
\begin{aligned}
s.t. \quad (T_{in,i}-T_{out,i})\,Fcp_i &= \sum_{k\in ST}\sum_{j\in CP} Q_{ijk} + Q_{CU,i}, && \forall i\in HP \\
(T_{out,j}-T_{in,j})\,Fcp_j &= \sum_{k\in ST}\sum_{i\in HP} Q_{ijk} + Q_{HU,j}, && \forall j\in CP \\
(T_{i,k}-T_{i,k+1})\,Fcp_i &= \sum_{j\in CP} Q_{ijk}, && \forall k\in ST,\ \forall i\in HP \\
(T_{j,k}-T_{j,k+1})\,Fcp_j &= \sum_{i\in HP} Q_{ijk}, && \forall k\in ST,\ \forall j\in CP \\
T_{in,i} &= T_{i,1}, && \forall i\in HP \\
T_{in,j} &= T_{j,NS+1}, && \forall j\in CP \\
T_{i,k} &\geq T_{i,k+1}, && \forall k\in ST,\ \forall i\in HP \\
T_{j,k} &\geq T_{j,k+1}, && \forall k\in ST,\ \forall j\in CP \\
T_{out,i} &\leq T_{i,NS+1}, && \forall i\in HP \\
T_{out,j} &\geq T_{j,1}, && \forall j\in CP
\end{aligned}
$$

$$
\begin{aligned}
(T_{i,NS+1} - T_{out,i})\, Fcp_i &= Q_{CU,i}, && \forall i \in HP \\
(T_{out,j} - T_{j,1})\, Fcp_j &= Q_{HU,j}, && \forall j \in CP \\
Q_{ijk} - \Omega z_{ijk} &\leq 0, && \forall k \in ST,\ \forall i \in HP, \\
& && \forall j \in CP \\
Q_{CU,i} - \Omega z_{CU,i} &\leq 0, && \forall i \in HP \\
Q_{HU,j} - \Omega z_{HU,j} &\leq 0, && \forall j \in CP \\
T_{i,k} - T_{j,k} + \Gamma(1 - z_{ijk}) &\geq \Delta T_{ijk}, && \forall k \in ST,\ \forall i \in HP, \\
& && \forall j \in CP \\
T_{i,k+1} - T_{j,k+1} + \Gamma(1 - z_{ijk}) &\geq \Delta T_{ijk+1}, && \forall k \in ST,\ \forall i \in HP, \\
& && \forall j \in CP \\
T_{i,NS+1} - T_{out,CU} + \Gamma(1 - z_{CU,i}) &\geq \Delta T_{CU,i}, && \forall i \in HP \\
T_{out,HU} - T_{j,1} + \Gamma(1 - z_{HU,j}) &\geq \Delta T_{HU,j}, && \forall j \in CP \\
\Delta T_{ijk} &\geq 10, && \forall k \in ST,\ \forall i \in HP, \\
& && \forall j \in CP \\
z_{ijk}, z_{CU,i}, z_{HU,j} &\in \{0,1\}, && \forall k \in ST,\ \forall i \in HP, \\
& && \forall j \in CP.
\end{aligned}
$$

Continuous Variables : The continuous variables are T_{ik}, the temperature of hot stream i at the hot end of stage k; T_{jk}, the temperature of cold stream j at the cold end of stage k; Q_{ijk}, the heat exchanged between hot stream i and cold stream j at temperature location k; $Q_{CU,i}$, the heat exchanged between hot stream i and the cold utility at temperature location k; $Q_{HU,j}$, the heat exchanged between cold stream j and the hot utility at temperature location k; ΔT_{ijk}, the temperature approach for the match of hot stream i and cold stream j at temperature location k; $\Delta T_{CU,i}$, the temperature approach for the match of hot stream i and the cold utility at temperature location k; $\Delta T_{HU,j}$, the temperature approach for the match of cold stream j and the hot utility at temperature location k.

Binary Variables : The binary variables are z_{ijk}, for the existence of a match between hot stream i and cold stream j at temperature location k; $z_{CU,i}$, for the existence of a match between hot stream i and the cold utility at temperature location k; $z_{HU,j}$, for the existence of a match between cold stream j and the hot utility at temperature location k.

Parameters : The parameters are T_{in}, the inlet temperature of a stream; T_{out}, the outlet temperature; Fcp, the heat capacity flowrate of a stream; Ω, the upper bound on heat exchange; Γ, the upper bound on the temperature difference. The stream data for the problem are summarized in Table 21.1. There are two temperature intervals. The steam utility cost is $C_{HU} = \$80$/kW-yr and the cooling water cost is $C_{CU} = \$15$/kW-yr. The fixed charges, CF, for the heat exchangers amount to \$5500/yr, regardless of the type of heat exchanger (hot process stream/cold utility, cold process stream/hot utility or hot process stream/cold process stream). The cost coefficient for the area-dependent part of the heat exchanger costs is $C = \$300$/yr. The overall heat transfer coefficients are $U_{ij} = 0.5$ kW/m^{2}K for the hot stream-cold stream units, $U_{CU,i} = 0.83333$ kW/m^{2}K for the cold stream-hot utility units and $U_{HU,j} = 0.5$ kW/m^{2}K for the hot stream-cold utility units.

The global optimum configuration involves six heat exchangers and is shown in Figure 21.5. It corresponds to an annualized network cost of \$154,997.

Remark 21.7.1 Due to the linear participation of the binary variables, the problem can be solved to global optimality using the SMIN-αBB algorithm. The area-dependent cost of the heat exchangers must be underestimated using the general convex lower bounding functions discussed in Section 21.2 (see equation (21.2)). The Outer Approximation algorithm is used to solve the underestimating convex MINLP at each node of the tree.

Study of Branching Strategies. The temperature variables are very important. Decreasing the range of one of the temperature variables results in a decrease of the range of some of the heat loads and stream temperature differences. The branching variable is therefore selected from the set of stream temperature variables exclusively, even though they do not participate explicitly in the nonconvex terms. The z_{ijk} binary variables representing the existence of hot stream/cold stream heat exchangers affect a larger number of variables than the $z_{CU,i}$ and $z_{HU,j}$ variables representing utility heat exchangers. They are therefore assigned a higher branching priority. Finally, to maximize the effect of branching, the bounds on the heat loads are expressed as a function of the bounds on the stream temperatures and the binary variables as shown below:

$$
\begin{aligned}
&0 \leq Q_{ijk} \leq z_{ijk}^{U} \ \min\left\{Fcp_i\left(T_{i,k}^{U} - T_{i,k+1}^{L}\right), Fcp_j\left(T_{j,k}^{U} - T_{j,k+1}^{L}\right)\right\}, \\
&\qquad\qquad \forall k \in ST,\ \forall i \in HP,\ \forall j \in CP, \\
&0 \leq Q_{CU,i} \leq z_{CU,i}^{U}\, Fcp_i\left(T_{i,NS+1}^{U} - T_{out,i}\right),\ \forall i \in HP, \\
&0 \leq Q_{HU,j} \leq z_{HU,i}^{U}\, Fcp_j\left(T_{out,j} - T_{j,1}^{L}\right), \forall j \in CP.
\end{aligned}
\tag{21.8}
$$

Similarly, the bounds on the temperature differences can be expressed in terms of the bounds on the branching variables.

$$
\begin{aligned}
&10 \leq \Delta T_{ij1} \leq 10\ \left(1 - z_{ij1}^U\right) + z_{ij1}^U\ \max\left\{T_{i,1}^U - T_{j,1}^L, 10\right\}, \forall i \in HP,\ \forall j \in CP, \\
&10 \leq \Delta T_{ijk} \leq 10\ \left(1 - z_{ijk-1}^U\right)\left(1 - z_{ijk}^U\right) \\
&\qquad + \left(1 - \left(1 - z_{ijk-1}^U\right)\left(1 - z_{ijk}^U\right)\right)\ \max\left\{T_{i,k}^U - T_{j,k}^L, 10\right\}, \\
&\qquad \forall i \in HP,\ \forall j \in CP, \forall 1 < k < NS, \\
&10 \leq \Delta T_{ijNS+1} \leq 10\ \left(1 - z_{ijNS}^U\right) + z_{ijNS}^U\ \max\left\{T_{i,NS+1}^U - T_{j,NS+1}^L, 10\right\}, \\
&\qquad \forall i \in HP,\ \forall j \in CP, \\
&10 \leq \Delta T_{CU,i} \leq 10\left(1 - z_{CU,i}^U\right) + z_{CU,i}^U\ \max\left\{T_{i,NS+1}^U - T_{out,CU}, 10\right\}, \forall i \in HP, \\
&10 \leq \Delta T_{HU,j} \leq 10\left(1 - z_{HU,j}^U\right) + z_{HU,j}^U\ \max\left\{T_{out,HU} - T_{j,1}^L, 10\right\}, \forall j \in CP.
\end{aligned}
\tag{21.9}
$$

All branching strategies proposed for the SMIN-αBB algorithm are tested for the solution of this problem. First, the continuous variables, that is, the temperatures, are branched on exclusively (Run 1a). Then, for Run 2a, the sequential branching approach is used. Since there are twelve binary variables, the first twelve levels of the tree may be explored by binary variable branching, corresponding to a maximum of 4,096 nodes generated through binary variable branching. Finally, the hybrid strategy is used for Runs 3a and 4a. In this case, a binary variable z is defined as close to its bounds if its value z^* at the solution of the relaxed MINLP is such that $\min\{z^*, 1 - z^*\} \leq zdist$. For Run 3a, $zdist = 0.1$ and for Run 4a, $zdist = 0.2$.

The results shown in Table 21.2 are obtained by performing variable bound updates on all the temperature variables at every iteration using the optimization-based approach without the bound improvement constraint on the objective function. The bounds on the other continuous variables are automatically updated through Eqs. (21.8) and (21.9). Although the nonconvexities arise only from the continuous variables, the SMIN-αBB does not converge after 1000 iterations when the continuous variable branching strategy is used (Run 1a). At that point, the relative difference between the upper and lower bound on the global solution is 2.5 % and the lower bound increases at the slow rate. Branching on the binary variables yields much better results. A total of 509 convex MINLPs and 798 convex NLPs are solved when using sequential branching (Run 2a). As a result, the average time per iteration is cut in half. The rate of lower bound improvement is also much faster and convergence is achieved in only 655 iterations. Higher improvement rates are obtained when using the hybrid strategy with $zdist = 0.1$ and $zdist = 0.2$. The deviation of the binary variables from integrality at the solution of a continuous relaxation is therefore a good measure of their effect on the quality of the lower bounding problem.

However, in both hybrid branching runs, the majority of lower bounding problems are convex MINLPs. In addition, the identification of a good branching variable requires the local solution of a nonconvex NLP. Both these factors combine to increase the CPU requirement per iteration to twice that of Run 2a. Consequently, it appears that random selection of binary variables is a better overall strategy.

Study of Variable Bound Updates. A number of variable bound update strategies are also tested for this problem. In this section we present two series of runs while the complete analysis can be found in Adjiman *et al.* (1998d).

Table 21.3 shows results using the optimization-based approach without the bound improvement constraint for both continuous and binary variables. In all cases, the number of iterations decreases slightly but the CPU requirements increase. The combination of optimization-based bound updates for the continuous variables and interval-based bound updates for the binary variables therefore seems sufficient.

Table 21.4 shows the results for varying the number of bound update variables. In run series f, the number of temperature variables whose bounds are updated using the optimization-based approach with the bound improvement constraint is varied. The frequency of bounds updates is every three iterations. As expected, a reduced number of variable bound updates decreases the average iteration costs and increases the total number of iterations. The best results are obtained by updating four temperature variables.

Remark 21.7.2 Using the best overall run (Run 2f), the heat exchanger network synthesis problem is solved in 180 iterations and 315 CPU seconds. This best run uses the sequential branching strategy and bound updates every three iterations on four of the temperature variables and all of the binary variables. The continuous variable bounds are tightened using the optimization-based approach with the bound improvement constraint and the binary variable bounds are updated using the interval-based approach. The results presented highlight the trade-offs that must be made between the quality of the lower bounding problem and the time expended to generate it. They also demonstrate that the presence of binary variables should be exploited to improve the performance of the SMIN-αBB algorithm.

21.7.2.2 Examples 2 and 3 : Use of Arithmetic Mean Temperature Difference. The two following examples are taken from Zamora and Grossmann (1998a). The difference in the formulation is due to the expression used for the area of the heat exchangers, which is based on the arithmetic mean temperature difference rather than the logarithmic mean or the Chen approximation. In the present case, the area of a heater for cold stream j is therefore given by

$$\frac{C_{j,HU} Q_{HU,j}}{\frac{U_{HU,j}}{2}\left(\Delta T_{HU,j} + T_{in,HU} - T_{out,j}\right)}. \tag{21.10}$$

For a cooler for hot stream i, the area is approximated by

$$\frac{C_{i,CU}Q_{CU,i}}{\frac{U_{CU,i}}{2}\left(\Delta T_{CU,i} + T_{out,i} - T_{in,CU}\right)}. \tag{21.11}$$

Finally, for a heat exchanger between two streams i and j in interval k, it is given by

$$\frac{C_{ij}Q_{ijk}}{\frac{U_{ij}}{2}\left(\Delta T_{ijk} + \Delta T_{ijk+1}\right)}. \tag{21.12}$$

In addition, the assumption of no stream splitting is imposed by adding two new sets of linear constraints, namely

$$\sum_{j \in CP} z(i,j,k) = 1, \qquad \forall\, i \in HP,\ \forall k \in ST, \tag{21.13}$$

$$\sum_{i \in HP} z(i,j,k) = 1, \qquad \forall\, j \in CP,\ \forall k \in ST. \tag{21.14}$$

In the following examples, a new variable ΔT^{+}_{ijk} is introduced to replace $\Delta T_{ijk} + \Delta T_{ijk+1}$ in Eq.(21.12). Thus, the area is expressed as

$$\frac{C_{ij}Q_{ijk}}{\frac{U_{ij}}{2}\Delta T^{+}_{ijk}}, \quad \forall\, i \in HP,\ \forall\, j \in CP,\ \forall k \in ST, \tag{21.15}$$

and the new linear constraint

$$\Delta T^{+}_{ijk} = \Delta T_{ijk} + \Delta T_{ijk+1}, \quad \forall\, i \in HP,\ \forall\, j \in CP,\ \forall k \in ST, \tag{21.16}$$

is incorporated in the problem. Based on this transformation, all nonconvex terms in the problem are fractional, of the form x_1/x_2, and can therefore be underestimated using the convex envelope (see Chapters 2 and 12). Thus, the area term for a cooler, Eq. (21.11), is replaced by a new variable $w_{CU,i}$ such

that

$$
\begin{aligned}
w_{CU,i} \geq\ & \frac{C_{i,CU}Q^L_{CU,i}}{\frac{U_{CU,i}}{2}\left(\Delta T_{CU,i} + T_{out,i} - T_{in,CU}\right)} \\
& + \frac{C_{i,CU}Q_{CU,i}}{\frac{U_{CU,i}}{2}\left(\Delta T^U_{CU,i} + T_{out,i} - T_{in,CU}\right)} \\
& - \frac{C_{i,CU}Q^L_{CU,i}}{\frac{U_{CU,i}}{2}\left(\Delta T^U_{CU,i} + T_{out,i} - T_{in,CU}\right)} \qquad (21.17) \\
w_{CU,i} \geq\ & \frac{C_{i,CU}Q^U_{CU,i}}{\frac{U_{CU,i}}{2}\left(\Delta T_{CU,i} + T_{out,i} - T_{in,CU}\right)} \\
& + \frac{C_{i,CU}Q_{CU,i}}{\frac{U_{CU,i}}{2}\left(\Delta T^L_{CU,i} + T_{out,i} - T_{in,CU}\right)} \\
& - \frac{C_{i,CU}Q^U_{CU,i}}{\frac{U_{CU,i}}{2}\left(\Delta T^L_{CU,i} + T_{out,i} - T_{in,CU}\right)} \qquad (21.18)
\end{aligned}
$$

and similar equations are used for a heater. The area cost for a hot stream-cold stream heat exchanger, as shown in Eq. (21.15) is replaced by the new variable w_{ijk} such that

$$
w_{ijk} \geq \frac{C_{ij}Q^L_{ijk}}{\frac{U_{ij}}{2}\Delta T^+_{ijk}} + \frac{C_{ij}Q_{ijk}}{\frac{U_{ij}}{2}\Delta T^{+,U}_{ijk}} - \frac{C_{ij}Q^L_{ijk}}{\frac{U_{ij}}{2}\Delta T^{+,U}_{ijk}} \qquad (21.19)
$$

$$
w_{ijk} \geq \frac{C_{ij}Q^U_{ijk}}{\frac{U_{ij}}{2}\Delta T^+_{ijk}} + \frac{C_{ij}Q_{ijk}}{\frac{U_{ij}}{2}\Delta T^{+,L}_{ijk}} - \frac{C_{ij}Q^U_{ijk}}{\frac{U_{ij}}{2}\Delta T^{+,L}_{ijk}}. \qquad (21.20)
$$

Finally, in the examples studied here, branching is performed on the variables that appear in the fractional terms, therefore Eqs. (21.8-21.9) are not used to update the bounds on these variables.

Example 2. This corresponds to example 4 of Zamora and Grossmann (1998a). Two hot streams and two cold streams are considered in a network with three temperature intervals. There are 16 binary variables, 64 continuous variables and 108 linear constraints. There are 16 fractional terms to be underestimated, one for each heat exchanger. A minimum temperature approach of 1 K is used for this problem, providing a lower bound on the ΔT and ΔT^+ variables. The required stream data are listed in Table 21.5. The steam cost is C_{HU} = \$80/kW-yr, the cooling water cost is C_{CU} = \$20/kW-yr. The overall heat transfer coefficient for the process stream heat exchangers and the coolers is 0.8 kW/m^2-K, and for the heaters, it is 1.2 kW/m^2-K. The cost coefficients for heat exchangers and coolers are CF = \$6250/yr and C = \$83.26/yr; for the heaters, they are CF = \$6250/yr and C = \$99.91/yr. The global solution of

the problem has a network cost of \$74,711/yr and involves heat exchangers between hot stream H2 and the cooling water, streams H1 and C1 in temperature intervals 1 and 3, streams H1 and C2 in temperature interval 2, and streams H2 and C1 in temperature interval 2.

The number of iterations and the CPU time are reported for different branching and bound update strategies in Table 21.6. For this formulation, it is best not to branch on the binary variables at all. When the continuous branching variable selection is based on the term measure, the heat duty and temperature difference variables are branched on equally. When the variable measure is used instead, only the heat duties are selected and better performance is obtained. This problem benefits from a tight initial lower bound of \$50,741/yr. Thus, while tightening the variable bounds enables the generation of a slightly higher initial lower bound (\$54,597/yr), the computational expense of this operation is not justified by the decrease in number of iterations. The best performance is obtained without any variable bound updates, when the algorithm converges in 69 iterations and 82.8 CPU seconds.

Example 3. This problem is example 5 of Zamora and Grossmann (1998a). This time, three hot streams and two cold streams are considered over three temperature intervals. There are therefore 23 binary variables, 90 continuous variables and 147 linear constraints. The stream data for this problem are listed in Table 21.7. The overall heat transfer coefficients are computed based on the film heat transfer coefficients listed in the table. The steam cost is C_{HU} = \$110/kW-yr and the cooling water cost is C_{CU} = \$10/kW-yr. The area cost of all heat exchangers is based on coefficients CF = \$7400/yr and C = \$80/yr. A minimum approach temperature of 1^oC is imposed. The global optimum solution has an annualized network cost of \$82,043/yr. It involves heat exchangers between hot streams H1 and H2 and the cooling water, cold stream C2 and the steam utility, streams H3 and C2 in temperature interval 1, and streams H3 and C1 in temperature interval 3.

The results of several runs are shown in Table 21.8. The same trends are observed as for the previous example. When branching is performed on the continuous variables only, using the variable measure, and no bound updates are involved, the SMIN-αBB algorithm identifies the global optimum solution in only 577.6 CPU seconds.

Remark 21.7.3 Zamora and Grossmann (1998a) solved to global optimality Examples 2 and 3 using their hybrid branch and bound approach. They reported the computational effort for examples 2 and 3 to be 156 CPU seconds and 6 CPU hours respectively on a IBM RISC/6000 workstation.

Summary

In this chapter, we discussed the theoretical and algorithmic aspects of the the SMIN-αBB algorithm which is applicable to locating the global solution of certain classes of nonconvex MINLP models. Furthermore, computational

studies of the SMIN-αBB algorithm were presented for small problems taken from the literature as well as medium size problems that arise in the synthesis of heat exchanger networks. Extensive study of the branching and variable bound updating strategies was performed for the case studies in the heat exchanger network synthesis. Additional computational studies and detailed analysis can be found in Adjiman *et al.* (1998d), while the explicit mathematical models and solutions are provided in the recent book on test problems for local and global optimization methods by Floudas *et al.* (1999).

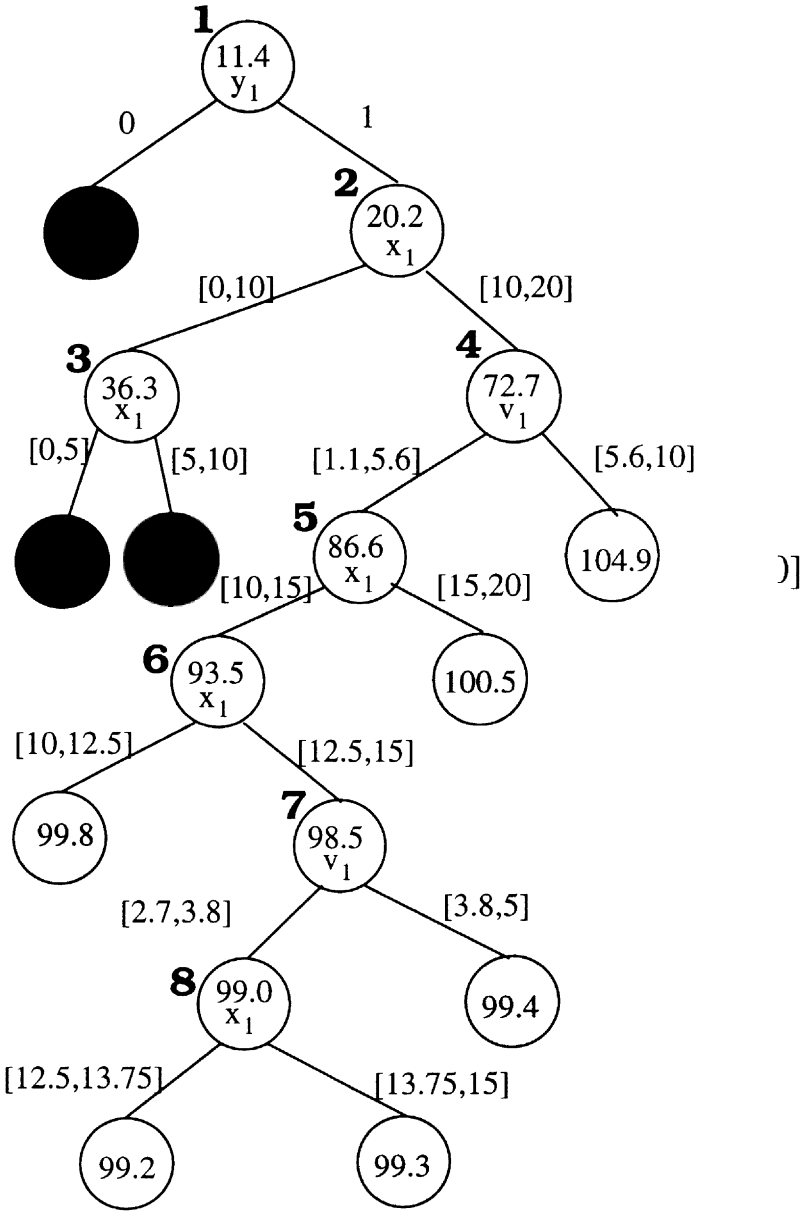


Figure 21.3. SMIN-αBB branch-and-bound tree for the illustrative example

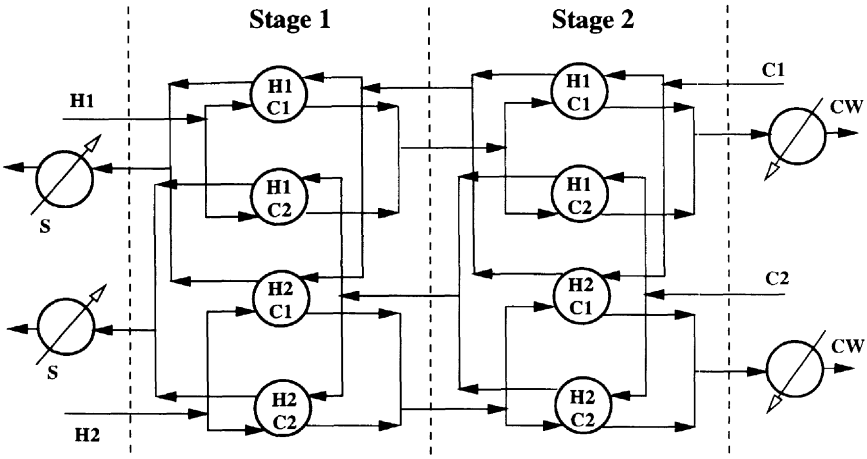


Figure 21.4. Superstructure for heat exchanger network Example 1.

Stream	T_{in} (K)	T_{out} (K)	Fcp (kW/K)
Hot 1	650	370	10.0
Hot 2	590	370	20.0
Cold 1	410	650	15.0
Cold 2	350	500	13.0
Steam	680	680	—
Water	300	320	—

Table 21.1. Stream data for heat exchanger network Example 1.

Run	Number of updated Ts	Update Frequency	Iterations	CPU sec	Deepest level	Binary branches
1a*	8	1	1000	2091	60	—
2a	8	1	655	644	38	317
3a	8	1	447	844	23	135
4a	8	1	445	812	28	158

Table 21.2. Heat exchanger network Example 1 results for four branching strategies with optimization-based bound updates on the T variables only, without the bound improvement constraint. * Run 1a has not converged after 1000 iterations.

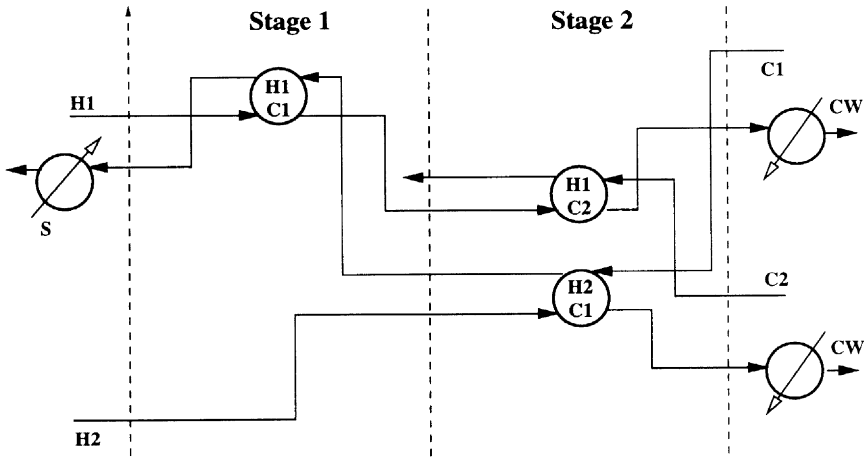


Figure 21.5. Optimum configuration for heat exchanger network Example 1.

Run	Number of updated Ts	Update Frequency	Iterations	CPU sec	Deepest level	Binary branches
1c*	8	1	1000	1823	62	—
2c	8	1	441	521	19	72
3c	8	1	287	644	28	93
4c	8	1	336	598	23	118

Table 21.3. Heat exchanger network Example 1 results for four branching strategies with optimization-based bound updates on the T and z variables. * Run 1c has not converged after 1000 iterations.

Run	Number of updated Ts	Update Frequency	Iterations	CPU sec	Deepest level	Binary branches
2f	6	3	148	532	16	81
2f	**4**	**3**	**180**	**315**	**20**	**88**
2f	2	3	443	376	21	208
3f	6	3	212	848	19	89
3f	4	3	263	626	16	89
3f	2	3	436	766	22	108
4f	6	3	222	802	17	109
4f	4	3	332	806	22	119
4f	2	3	618	715	29	123

Table 21.4. Heat exchanger network Example 1 results for four branching strategies with optimization-based bound updates with the bound improvement constraint on some of the T variables and interval-based bound updates on the z variables.

Stream	T_{in} (K)	T_{out} (K)	Fcp (kW/K)
H1	443	333	30
H2	423	303	15
C1	293	408	20
C2	353	413	40
Steam	450	450	—
Water	293	313	—

Table 21.5. Stream data for Example 2.

Branching strategy	Bound update strategy	Number of iterations	CPU time (seconds)
Sequential and variable measure	No bound updates	465	250.6
Continuous only with variable measure	**No bound updates**	**69**	**82.8**
Continuous only with term measure	No bound updates	77	98.2
Continuous only with variable measure	First iteration	35	109.3
Continuous only with variable measure	Every iteration	21	800.5

Table 21.6. Results for Example 2.

Stream	T_{in} (^oC)	T_{out} (^oC)	Fcp (kW/^oC)	h (kW/(m^2 ^oC))
H1	159	77	2.285	0.10
H2	267	80	0.204	0.04
H3	343	90	0.538	0.50
C1	26	127	0.933	0.01
C2	118	265	1.961	0.50
Steam	300	300	—	0.05
Water	20	60	—	0.20

Table 21.7. Stream data for Example 3.

Branching strategy	Bound update strategy	Number of iterations	CPU time (seconds)
Sequential and variable measure	No bound updates	1699	2191.2
Continuous only with variable measure	**No bound updates**	**65**	**577.6**
Continuous only with term measure	No bound updates	1034	14818.3
Continuous only with variable measure	First iteration	39	893.5
Continuous only with variable measure	Every iteration	31	1110.8

Table 21.8. Results for Example 3.

22 THE GMIN-αBB APPROACH : THEORY AND COMPUTATIONS

This chapter presents the General structure Mixed Integer Nonlinear αBB global optimization approach, GMIN-αBB, for general nonconvex MINLP problems. Section 22.1 describes the mathematical models that the GMIN-αBB can address. Section 22.2 focuses on the generation of valid lower bounds. Section 22.3 discusses the generation of upper bounds. Section 22.4 outlines the strategies for the selection of the branching variables. Section 22.5 describes the updates of the variable bounds through an optimization-based approach and an interval analysis based approach. Section 22.6 presents the algorithmic steps of the GMIN-αBB approach and a small illustrative example. Finally, in section 22.7 computational studies are reported on (i) small literature problems, (ii) pump configuration problems, and (iii) trim loss optimization problems. The material of this chapter is based on the work of Adjiman *et al.* (1997c), (1998d).

C. A. Floudas, *Deterministic Global Optimization*
© Springer Science+Business Media Dordrecht 2000

22.1 MATHEMATICAL FORMULATION

The General structure Mixed Integer Nonlinear αBB, GMIN-αBB, algorithm guarantees ϵ-convergence to a global solution of the following classes of MINLPs:

$$\begin{aligned} \min_{\boldsymbol{x},\boldsymbol{y}} \quad & f(\boldsymbol{x},\boldsymbol{y}) \\ s.t. \quad & \boldsymbol{g}(\boldsymbol{x},\boldsymbol{y}) \leq \boldsymbol{0} \\ & \boldsymbol{h}(\boldsymbol{x},\boldsymbol{y}) = \boldsymbol{0} \\ & \boldsymbol{x} \in [\boldsymbol{x}^L, \boldsymbol{x}^U] \\ & \boldsymbol{y} \in [\boldsymbol{y}^L, \boldsymbol{y}^U] \cap \mathcal{N}^q \end{aligned} \tag{22.1}$$

where $f(\boldsymbol{x},\boldsymbol{y})$, $\boldsymbol{g}(\boldsymbol{x},\boldsymbol{y})$ and $\boldsymbol{h}(\boldsymbol{x},\boldsymbol{y})$ are functions whose continuous relaxation is twice continuously differentiable, q is the number of integer variables and $\mathcal{N}$ is the set of integers.

The GMIN-αBB algorithm extends the applicability of the standard branch-and-bound approaches for MINLPs (Beale, 1977; Gupta and Ravindran, 1985; Ostrovsky *et al.*, 1990; Borchers and Mitchell, 1991; Quesada and Grossmann, 1992) by making use of the αBB algorithm. The solution space is explored by branching on at least one integer variable at each node. A lower bound on the solution of the problem within a given region is obtained by solving a continuous relaxation of the mixed-integer problem. The validity of the lower bound and hence the optimality properties of the final solution depend on the ability to identify the global optimal solution of the relaxation. In the case of an MILP, this relaxation is an LP which can be solved to global optimality using the simplex algorithm or an interior point method. For an MINLP whose continuous relaxation is convex, a standard NLP solver such as MINOS (Murtagh and Saunders, 1983) can identify a global solution of the relaxation. In the case of the general problems represented by (22.1), the generation of a valid lower bound is discussed in the next section.

22.2 GENERATION OF LOWER BOUNDS

The GMIN-αBB algorithm is a branch and bound global optimization method for very broad classes of MINLPs stated in (22.1), and it is based on the αBB principles. The αBB global optimization algorithm for twice continuously differentiable NLPs is used to obtain a guaranteed lower bound on the global solution at the current node of the branch-and-bound tree, as shown in Figure 22.1. At every iteration of the αBB algorithm, a lower bounding problem for the nonconvex NLP is formulated. Depending on the GMIN-αBB node being explored, one of two approaches can be used when running the αBB algorithm to get a lower bound on the continuous relaxation :

Case 1 : If at least one integer variable can be relaxed at the current node, run the αBB algorithm for a few iterations to obtain a valid lower bound on

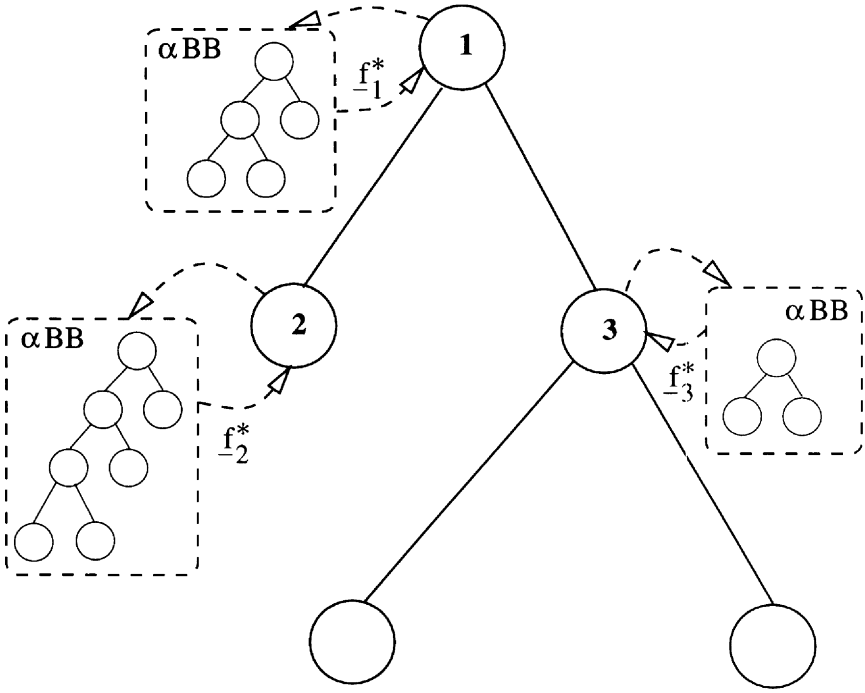


Figure 22.1. Branch-and-Bound Tree for the GMIN-αBB.

the global solution of the continuous relaxation *or* run the αBB algorithm to completion to obtain the global solution of the continuous relaxation.

Case 2 : Otherwise, run the αBB algorithm to completion to obtain the global solution for the current node.

This strategy makes use of the fact that the rate of improvement of the lower bound on the global solution of a nonconvex NLP is usually high at early iterations of the αBB algorithm and then gradually tapers off. The number of iterations required to reach the slow convergence phase depends on the characteristics of the NLP. To illustrate this point, the relative quality ρ of the lower bound on the objective function at iteration i, $\underline{f}_i^{NLP}$, can be computed as $\rho = (\underline{f}_i^{NLP} - \underline{f}_0^{NLP})/(f^{NLP,*} - \underline{f}_0^{NLP})$, where $f^{NLP,*}$ is the global solution. The relative quality ρ is plotted in Figure 22.2 for five nonconvex continuous relaxations of the pump network synthesis problem presented in Section 22.7.2. In each case, some of the integer variables are fixed to the values indicated on the figure and the αBB is run for forty iterations. For $z = (0, 0, 1)$, the lower bound improves significantly over the first 30 iterations, while for $z = (1, 1, 0)$, rapid progress is seen for only 15 iterations.

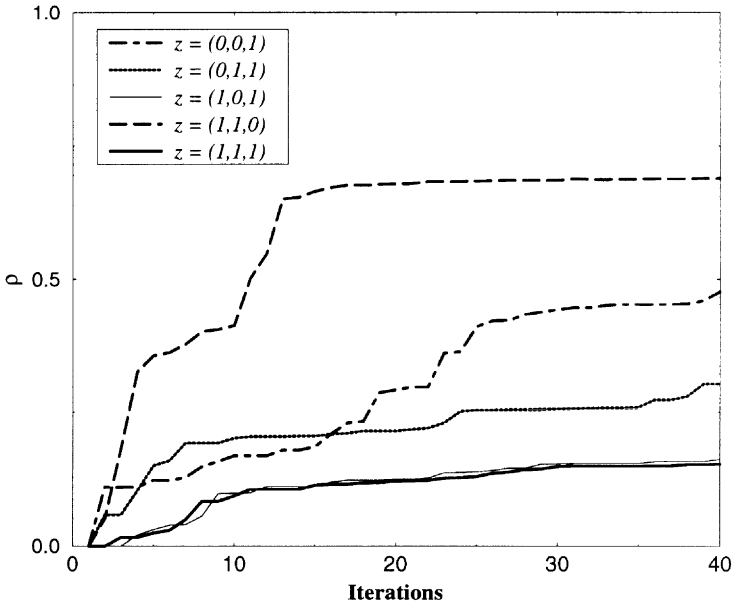


Figure 22.2. Progress of ρ, the relative quality of the lower bound, for five continuous relaxations.

To maximize the lower bound improvement rate for the GMIN-αBB algorithm, the αBB algorithm should not be run in the regions of slow convergence. The number of αBB iterations used in Case 1, that is, when not all integer variables are fixed at the current GMIN-αBB node, should therefore be limited to a number that is sufficiently large to take advantage of the rapid gain in quality of the lower bound, but that reduces the amount of time spent solving a slowly converging NLP. Ideally, this number should be dependent on the characteristics of the NLP being solved. An adaptive rule that monitors the progress of the αBB run was proposed by Adjiman *et al.* (1998d). It aims to verify that over the last m iterations, where m is a user-specified quantity, the convergence of the algorithm has at least improved by some fraction r. Let $\overline{f}_i^{NLP}$ be the best known upper bound on the solution of the NLP at iteration i and let $\underline{f}_i^{NLP}$ be the lower bound at iteration i. Then, the αBB run is allowed to proceed to iteration $i+1$, $i \geq m$, as long as

$$\frac{\left(\overline{f}_i^{NLP} - \underline{f}_i^{NLP}\right) - \left(\overline{f}_{i-m}^{NLP} - \underline{f}_{i-m}^{NLP}\right)}{\overline{f}_{i-m}^{NLP} - \underline{f}_{i-m}^{NLP}} \geq r. \tag{22.2}$$

Typically, m can be set to 3 iterations and r to 30%.

In order to ensure that this bounding procedure is consistent (Horst and Tuy, 1993) and that convergence of the algorithm can be guaranteed, the lower bound at a child node must be at least as large as the lower bound for its parent. If the continuous relaxation is solved to global optimality at every node, this condition is always satisfied. When the αBB run is terminated before convergence, however, the stopping criterion may be satisfied at a lower value of the lower bound for a child node than for its parent. Thus, if the lower bound on a parent node is $\underline{f}_l$, the stopping criterion must only be imposed on the child node *after* its lower bound $\underline{f}_{l+1}$ is such that $\underline{f}_{l+1} \geq \underline{f}_l$. A procedure that can generate a lower bound on a nonconvex MINLP without running the αBB algorithm to completion can be found in Adjiman *et al.* (1998d).

Remark 22.2.1 An effective way to reduce the computational expense of αBB runs is to make use of the known best upper bound on the global optimum solution of the nonconvex MINLP, $\overline{f}^*$. This can either be incorporated in the NLP as a constraint that specifies an upper bound on the objective function, or it can be added as a stopping criterion for the NLP algorithm. This latter option was chosen in the implementation.

Remark 22.2.2 The use of continuous relaxation to solve a mixed-integer problem may result in a large integrality gap, especially at early nodes in the branch-and-bound tree. In order to narrow this gap, it is important to obtain a lower bounding problem that preserves the discrete nature of the problem as much as possible. One way to enforce this is to identify any *binary* variables in the problem which participate only in linear or mixed-bilinear terms. If such variables exist, they do not need to be relaxed to formulate the lower bounding problem. The partial continuous relaxation that results from this approach is a nonconvex MINLP of type (21.1) which can be solved to global optimality using the SMIN-αBB algorithm presented in Chapter 21. The overall lower bounding strategy remains the same, but the αBB algorithm is replaced by the SMIN-αBB algorithm. For simplicity, the remaining sections describing the GMIN-αBB algorithm assume that the continuous relaxation is a nonconvex NLP.

22.3 GENERATION OF UPPER BOUNDS

Since branching occurs only on integer variables, the size of the branch-and-bound tree is finite and it is not necessary to generate an upper bound on the nonconvex MINLP at each node in order to guarantee convergence of the GMIN-αBB algorithm. However, upper bounds play a significant role in improving the convergence rate of the algorithm by allowing the fathoming of nodes whose lower bound is greater than the best known upper bound and therefore reducing the final size of the branch-and-bound tree. An upper bound on the solution of a given node can be obtained in several ways. For example, if the solution of the continuous relaxation is integer-feasible, this solution is both a lower and an upper bound on the current node. If the αBB algorithm

was run for only a few iterations and the relaxed integer variables are integer at the lower bound, they can be fixed to these integer values and the resulting nonconvex NLP can be solved locally to yield an upper bound on the solution of the node. Finally, a set of integer values satisfying the integer constraints can be used to construct a nonconvex NLP whose local solutions are upper bounds on the current node solution.

22.4 SELECTION OF BRANCHING VARIABLES

The bisection of a variable range takes place either at the midpoint of the range of the selected variable, or at the value of that variable at the solution of the lower bounding problem. Since branching is performed on integer variables, the lower and upper bounds on the range are rounded respectively to the nearest smaller and larger integers. It is also possible to branch on more than one variable at a given node, or to perform k-section on one of the variables. An integer variable is chosen randomly or according to branching priorities. An additional rule consists of selecting the most or least fractional variable at the solution of a continuous relaxation of the problem.

22.5 UPDATES OF VARIABLE BOUNDS

The same general strategies are used for variable bound tightening as in the SMIN-αBB algorithm presented in Chapter 21, namely an optimization-based or an interval-based approach. By taking advantage of the integrality of the $\boldsymbol{y}$ variables, these two techniques can be extended to update bounds on integer variables.

22.5.1 Optimization-based approach

In the optimization approach, the lower or upper bound on variable y_i is improved by first relaxing the integer variables, and then solving the convex NLP

$$y^* = \begin{cases} \min\limits_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{w}} \,/\, \max\limits_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{w}} & y_i \\ \quad s.t. & \breve{f}(\boldsymbol{x},\boldsymbol{y},\boldsymbol{w}) \leq \overline{f}^* \\ & \breve{\boldsymbol{g}}(\boldsymbol{x},\boldsymbol{y},\boldsymbol{w}) \leq \boldsymbol{0} \\ & \breve{h}_i(\boldsymbol{x},\boldsymbol{y},\boldsymbol{w}) = 0, \; i \in P_L \\ & \breve{h}_i(\boldsymbol{x},\boldsymbol{y},\boldsymbol{w}) = \; 0, \; i \in P_N \\ & \breve{h}_i^{-}(\boldsymbol{x},\boldsymbol{y},\boldsymbol{w}) = \; 0, \; i \in P_N \\ & \mathcal{C}(\boldsymbol{x},\boldsymbol{y},\boldsymbol{w}) \leq \boldsymbol{0} \\ & \boldsymbol{x} \in [\boldsymbol{x}^L, \boldsymbol{x}^U] \\ & \boldsymbol{y} \in [\boldsymbol{y}^L, \boldsymbol{y}^U] \\ & \boldsymbol{w} \in [\boldsymbol{w}^L, \boldsymbol{w}^U] \end{cases} \tag{22.3}$$

where the $\breve{}$ superscript denotes a convex underestimator of the specified function valid for the domain $(\boldsymbol{x},\boldsymbol{y}) \in [\boldsymbol{x}^L, \boldsymbol{x}^U] \times [\boldsymbol{y}^L, \boldsymbol{y}^U]$ and the remaining notation is as for problem (21.6). The final updated lower or upper bound on the integer variable is obtained by setting $y_i^L = \lceil y^* \rceil$ or $y_i^U = \lfloor y^* \rfloor$.

22.5.2 *Interval-based approach*

In the interval-based approach, an iterative procedure is followed based on an interval test which provides necessary conditions for the feasibility of the original constraints and the bound improvement constraint $f(\boldsymbol{x},\boldsymbol{y}) \leq \overline{f}^*$, given the relaxed region $(\boldsymbol{x},\boldsymbol{y}) \in [\boldsymbol{x}^L, \boldsymbol{x}^U] \times [\boldsymbol{y}^L, \boldsymbol{y}^U]$. The approach is similar to that followed in Section 21.5.2. The set of constraints defines a region $\mathcal{F} = \{(\boldsymbol{x},\boldsymbol{y}) : f(\boldsymbol{x},\boldsymbol{y}) \leq \overline{f}^*; \boldsymbol{g}(\boldsymbol{x},\boldsymbol{y}) \leq 0; \boldsymbol{h}(\boldsymbol{x},\boldsymbol{y}) = 0; (\boldsymbol{x},\boldsymbol{y}) \in [\boldsymbol{x}^L, \boldsymbol{x}^U] \times [\boldsymbol{y}^L, \boldsymbol{y}^U]\}$. The modified procedure to improve the lower (upper) bound on variable y_i ensures that the integrality of the variable bounds is maintained at every iteration.

22.5.3 *Lower bound improvement*

The variable bound tightening procedures are applied before the αBB algorithm is entered to generate a lower bound on the current node. In many cases, during an αBB run, variable bound updates are also used to improve the quality of the lower bounding problem. Although the αBB algorithm treats the $\boldsymbol{y}$ variables as continuous, the bound update strategy within the αBB algorithm may be modified to account for the true nature of these variables. A larger reduction in the solution space can be achieved by adopting one of the *integer*

bound update strategies described here for the relaxed $\boldsymbol{y}$ variables. This more restrictive approach leads to a lower bound which is not necessarily a valid lower bound on the continuous relaxation, but which is always a lower bound on the global solution of the nonconvex MINLP at the current node. The theoretical limitation of the integrality gap is thus removed. The minimum achievable distance between the global solution of the nonconvex MINLP at the current node and the generated lower bound cannot be predicted, but it is smaller than or equal to the integrality gap.

Further progress can be made by modifying the branching step in the αBB algorithm. Whenever a relaxed integer variable is branched on, the bounds on that variable should be rounded to the nearest integer. Thus, if branching takes place through a bisection at the midpoint for variable $y \in [y^L, y^U]$, the two new regions should correspond to $y \in [y^L, \lfloor y^M \rfloor]$ and $y \in [\lceil y^M \rceil, y^U]$, with $y^M = \frac{y^U + y^L}{2}$.

22.6 THE GMIN-αBB ALGORITHM

The upper and lower bound generation schemes, the branching strategies for integer variables and the bound update techniques are combined to produce the GMIN-αBB algorithm. The main steps of the algorithm are shown in Figure 22.3.

22.6.1 Illustrative example

The algorithmic procedure for the GMIN-αBB algorithm is illustrated using the same example as for the SMIN-αBB algorithm presented in Chapter 21. The branch-and-bound tree is shown in Figure 22.4 using the same notation as previously. At the first node, the continuous relaxation of the nonconvex MINLP is solved using ten αBB iterations to yield a lower bound of 48.6. No upper bound is found. Next, the binary variable y_2 is chosen for branching and the continuous relaxation of the problem with $y_2 = 0$ is solved. A lower bound of 92.2 is found as the global solution to this nonconvex NLP. The variable y_1 is equal to 1 at this solution which therefore provides an upper bound on the global optimum solution of the nonconvex MINLP. The region $y_2 = 1$ is then examined and the lower bound on the NLP is found to be 97.0 at the sixth αBB iteration. Since this is greater than the best upper on the global solution of the MINLP, the αBB run is terminated and this node can be fathomed. The bounds on the solution are within the pre-specified tolerance at the left child of node 1 and convergence has therefore been achieved.

22.7 COMPUTATIONAL STUDIES

In this section, we discuss several computational studies of the GMIN-αBB for (i) small literature problems, (ii) pump network problem, and (iii) four trim loss minimization problems. For the solution of convex MINLPs, the GMIN-αBB algorithm relies on the implementations of the Generalized Benders Decompo-

sition and the Outer Approximation which are part of MINOPT (Schweiger and Floudas, 1997).

22.7.1 Small Literature Examples

The examples studied here are the same ones discussed in section 21.7.1 of Chapter 21 where the mathematical formulations are provided, and all runs are performed on an HP-C160.

Example 1. The value of the objective function at the global solution is 7.67 and the optimum solution vectors are $\boldsymbol{x}^* = (1.12, 1.31)^T$ and $\boldsymbol{y}^* = (0, 1, 1)^T$. The GMIN-$\alpha$BB algorithm solves the problem at the root node in 0.14 CPU seconds.

Example 2. The global solution has an objective value of 1.07654 and the corresponding solution vectors are $\boldsymbol{x}^* = (0.94194, -2.1)^T$ and $y^* = 1$. The GMIN-αBB algorithm converges after two iterations and less than 0.1 CPU second. The GMIN-αBB algorithm explores three nodes: the root node and its two children, $y = 0$ and $y = 1$.

Example 3. The global solution has an objective value of 4.5796 and solution vectors $\boldsymbol{x}^* = (0.2, 0.8, 1.908)^T$ and $\boldsymbol{y}^* = (1, 1, 0, 1)^T$. The GMIN-$\alpha$BB algorithm requires three iterations and 0.7 CPU seconds.

Example 4. The global optimum solution consists of an objective function value of -0.94347 and solution vectors : $\boldsymbol{x}^* = (0.97, 0.9925, 0.98)^T$, and $\boldsymbol{y}^* = (0, 1, 1, 1, 0, 1, 1, 0)^T$. The GMIN-$\alpha$BB algorithm converges after eighteen iterations and 4.7 CPU seconds with the original formulation. The αBB algorithm is run for only three iterations at every mixed-integer node. The GMIN-αBB algorithm can also be used to solve the reformulated problem. In this case, the solution is found after ten iterations and 3.0 CPU seconds.

Example 5. The global optimum solution is 31, with $(y_1, y_2) = (3, 1)^T$. The original formulation is solved in four iterations and 0.9 CPU seconds with the GMIN-αBB algorithm. The binary reformulation is solved in six iterations and 2.2 CPU seconds using the GMIN-αBB algorithm.

Example 6. The optimal solution is -17, with $x^* = 4$ and $y^* = 1$. The GMIN-αBB algorithm solves this problem at the root node in 0.7 CPU seconds. The reformulation in terms of binary variables is solved at the root node in 0.7 CPU seconds by the GMIN-αBB algorithm.

22.7.2 Pump Network Synthesis

In this example, taken from Westerlund *et al.* (1994), the aim is to identify the least costly configuration of centrifugal pumps that achieves a pre-specified

pressure rise based on a given total flowrate. The problem belongs to the class of nonconvex MINLP models that can be addressed by the GMIN-αBB algorithm: it involves integer variables that participate in highly nonconvex terms. Figure 22.5 shows a schematic representation of a three level pump configuration, where each level corresponds to a different pump type. Figure 22.6 shows the global optimal configuration, corresponding to an annualized cost of 128,894 FIM (FIM stands for Finmark).

The structural decisions for an L level superstructure are represented by a number of discrete variables. The binary variables z_i, $i = 1, \ldots, L$, denote the existence of level i. The integer variables Np_i, $i = 1, \ldots, L$, denote the number of parallel lines at level i ($Np_i \in \{0, \ldots, NP\}$, where NP is the maximum number of parallel lines). The integer variables Ns_i, $i = 1, \ldots, L$, denote the number of pumps in series at level i ($Ns_i \in \{0, \ldots, NS\}$, where NS is the maximum number of pumps in series). The relevant continuous variables are the fraction of total flow going to level i, x_i, the flowrate on each line at level i, $\dot{v}_i$, the rotation speed of all pumps on level i, ω_i, the power requirements at level i, P_i, and the pressure rise at level i, Δp_i. For a three-level superstructure, the minimization of the annualized cost is given by :

$$\min \sum_{i=1}^{3} (C_i + C_i' P_i) Np_i Ns_i z_i$$

$$\begin{aligned}
s.t. \quad & P_1 - 19.9(\tfrac{\omega_1}{\omega_{max}})^3 - 0.1610(\tfrac{\omega_1}{\omega_{max}})^2 \dot{v}_1 + 0.000561(\tfrac{\omega_1}{\omega_{max}})\dot{v}_1^2 = 0 \\
& P_2 - 1.21(\tfrac{\omega_2}{\omega_{max}})^3 - 0.0644(\tfrac{\omega_2}{\omega_{max}})^2 \dot{v}_2 + 0.000564(\tfrac{\omega_2}{\omega_{max}})\dot{v}_2^2 = 0 \\
& P_3 - 6.52(\tfrac{\omega_3}{\omega_{max}})^3 - 0.1020(\tfrac{\omega_3}{\omega_{max}})^2 \dot{v}_3 + 0.000232(\tfrac{\omega_3}{\omega_{max}})\dot{v}_3^2 = 0 \\
& \Delta p_1 - 629(\tfrac{\omega_1}{\omega_{max}})^2 - 0.696(\tfrac{\omega_1}{\omega_{max}})\dot{v}_1 + 0.0116\dot{v}_1^2 = 0 \\
& \Delta p_2 - 215(\tfrac{\omega_2}{\omega_{max}})^2 - 2.950(\tfrac{\omega_2}{\omega_{max}})\dot{v}_2 + 0.0115\dot{v}_2^2 = 0 \\
& \Delta p_3 - 361(\tfrac{\omega_3}{\omega_{max}})^2 - 0.530(\tfrac{\omega_3}{\omega_{max}})\dot{v}_3 + 0.00946\dot{v}_3^2 = 0 \\
& x_1 + x_2 + x_3 = 1 \\
& \omega_i - \omega_{max} \leq 0, \; i = 1, 2, 3 \\
& \dot{v}_i - \tfrac{x_i}{Np_i} V_{tot} = 0, \; i = 1, 2, 3 \\
& \Delta P_{tot} - \Delta p_i Ns_i = 0, \; i = 1, 2, 3
\end{aligned}$$

	Pump 1	Pump 2	Pump 3
Capital cost (FIM)	38,900	15,300	20,100
a	629.0	215.0	361.0
b	0.696	2.950	0.530
c	-0.01160	-0.11500	-0.00946
α	19.9	1.21	6.52
β	0.1610	0.0644	0.1020
γ	-0.000561	-0.000564	-0.000232

Table 22.1. Data for the pump network synthesis problem.

$$
\begin{aligned}
Np_i &= \{0,1,2,3\},\ i = 1,2,3 \\
Ns_i &= \{0,1,2,3\},\ i = 1,2,3 \\
z_i &= \{0,1\},\ i = 1,2,3.
\end{aligned}
\tag{22.4}
$$

The parameter values are $V_{tot} = 350$ m^3/h, $\Delta P_{tot} = 400$ kPa and $\omega_{max} = 2950$ rpm. The costs and manufacturer specifications for the pumps are given in Table 22.1. The capital costs are annualized by multiplying the fixed pump costs by a factor of 0.1627 which corresponds to a 10% interest rate and a 10 year life. The operating costs are based on 6000 hours per year and an electricity cost of 0.3 FIM/kWh.

Formulation (22.4) is nonconvex in the continuous and integer variables. Thirty seven local minima have previously been reported for this problem (Westerlund *et al.*, 1994). Considering the range of values of the integer variables, over 32,000 combinations could be obtained.

Some of the integer variables contribute more than others to the complexity of the problem. Specifically, the binary variables, z_i, associated with the existence or non-existence of a level, are important because they help reduce the size of the NLP problem by setting the variables associated with a particular level to zero when that level does not exist. The following additional linear constraints can reflect this fact.

$$
P_i \le z_i\, P_{max,i}, \quad \Delta p_i \le z_i\, \Delta p_{max,i}, \quad \dot{v}_i \le z_i\, V_{tot}, \quad x_i \le z_i,
$$

$$
\omega_i \le z_i\, \omega_{max}, \quad Np_i \le z_i\, NP, \quad Ns_i \le z_i\, NS.
$$

where $P_{max} = (145, 93, 71)^T$ and $\Delta p_{max} = (1807, 592, 1585)^T$.

Different strategies of the GMIN-αBB algorithm are tested for lower bound generation, branching variable selection and variable bound tightening. These are discussed in the following.

Maximum αBB iterations	GMIN-αBB nodes	CPU seconds
2	969	4710
10	991	8478
20	775	7878
100	799	27464

Table 22.2. Results for the pump network synthesis problem varying the maximum number of αBB iterations, using the most fractional variable branching variable selection and no bound updates.

Lower Bound Generation. Because the relaxed NLP formulation is highly nonlinear, the αBB algorithm is not run to completion unless all integer variables are fixed. As discussed in Section 22.2, a balance is sought between the computational cost of generating a lower bound and its proximity to the global solution of the relaxation. The problem is first solved by imposing different limits on the maximum number of αBB iterations, $iter_{max}$. Along the GMIN-αBB tree, the most fractional variable at the solution of the lower bounding problem is selected for branching and no bound updates are performed. When the αBB algorithm is entered, the bounds on all variables are updated once and the third branching strategy (see Chapter 12) is used to select a continuous or relaxed integer variable. The results are shown in Table 22.2. The average time requirements per node when $iter_{max}$ is two iterations only are seven times smaller than when it is one hundred. The stopping criterion used for the αBB algorithm includes the maximum number of iterations, $iter_{max}$, the upper bound on the global optimum solution of the nonconvex MINLP and the lower bound on the relaxation of the parent node. Consequently, the computational effort for the generation of a lower bound differs at each GMIN-αBB node. When $iter_{max} = 2$, for instance, $iter_{max}$ serves as a termination criterion at 471 nodes of the GMIN-αBB tree out of the total 969 nodes. Only one αBB iteration is required at 433 other nodes. This means that, at the first iteration, the underestimating problem is found to be infeasible or its solution is greater than the upper bound on the global optimum solution of the MINLP. At the remaining 65 nodes, all integer variables are fixed and $iter_{max}$ can no longer be used as a stopping criterion. These nodes require between 2 and 219 iterations. For $iter_{max} = 20$, the maximum number of αBB iterations is used as a stopping criterion for 403 nodes out of a total of 775. For 310 other GMIN-αBB nodes, only one αBB iteration is needed. The remaining 62 nodes require a wide array of run lengths, with 36 nodes involving less than twenty αBB iterations.

Branching Strategy. Because of the dependence of the Ns_i and Np_i variables on z_i, a substantial portion of the branch-and-bound tree can be fathomed

Maximum αBB iterations	GMIN-αBB nodes	CPU seconds
2	835	2123
10	721	3729
20	547	2515
100	455	8428

Table 22.3. Results for the pump network synthesis problem varying the maximum number of αBB iterations, using prioritized branching variable selection and no bound updates.

Maximum αBB iterations	GMIN-αBB nodes	CPU seconds
2	153	1036
10	137	1024
20	139	1716
100	113	3707

Table 22.4. Results for the pump network synthesis problem varying the maximum number of αBB iterations, using prioritized branching variable selection and bound updates at every iteration.

if a particular combination of the z_i variables is excluded early on. This observation suggests a branching scheme in which high priorities are assigned to the three binary variables defining the existence or non-existence of a level. After the third level of the GMIN-αBB tree, the remaining integer variables are branched on based on the most fractional variable rule. The results for this strategy using different values of $iter_{max}$ are shown in Table 22.3. This strategy has a very beneficial effect on all runs, as the number of nodes decreases by up to 43% and the CPU requirements are reduced by up to 70%. When $iter_{max} = 20$, for instance, the computational effort per node changes from 10.2 seconds to 5.0 seconds. This is in part due to the fact that the total number of αBB iterations is 7565 with prioritized branching, as opposed to 9027 with most fractional variable branching. The generation of tighter convex relaxations that results from fixing important variables first permits the fast elimination of an increased number of nodes.

Variable Bound Updates. The bounds on the integer variables can be updated at every iteration of the GMIN-αBB algorithm using the optimization and interval based approaches described in Section 22.5. The results using the optimization-based procedure are shown in Table 22.4. When the problem

m	r	GMIN-αBB nodes	CPU seconds	αBB iterations Average	Maximum	Total
1	0.05	167	943	2.9	137	479
1	0.1	149	976	11.9	367	1767
1	0.3	147	976	9.9	388	1455
3	**0.1**	**147**	**927**	**5.3**	**124**	**781**
3	0.3	149	960	9.7	192	981
7	0.1	149	1046	12.4	379	1844
7	0.3	139	987	11.1	285	1549
15	0.1	139	1215	19.4	263	2690
15	0.3	141	1163	18.2	363	2571

Table 22.5. Results for the pump network synthesis problem using automatic stopping criteria for the αBB runs, prioritized variable branching and optimization-based bound updates.

is solved with interval-based bound updates, this strategy does not result in a significant reduction of the variable ranges. For two αBB iterations, for example, 899 GMIN-αBB nodes are explored and over 10000 CPU seconds are required.

Automatic αBB Termination. As discussed in Section 22.2, the ideal value of $iter_{max}$, the maximum number of αBB iterations, depends on the structure of the problem. The results shown in Table 22.5 are obtained by imposing the test of Eq. (22.2), with different values of m and r. The first conclusion that may be drawn from this table is that the automatic stopping criterion leads to runs of the same quality as a user-specified maximum number of iterations. Overall, the behavior of the algorithm seems quite robust to changes in the values of m and r. Detailed analysis of the automatic termination criterion can be found in Adjiman *et al.* (1998d).

22.7.3 Trim Loss Minimization

Trim loss minimization problems arise in the paper cutting industry and have been studied in the context of integer nonconvex optimization by Westerlund *et al.* (1994) and Harjunkoski (1997). The main task is to cut out some paper products of different sizes from a large raw paper roll, in order to meet a customer's order. Each product paper roll is characterized by its width b. All product rolls are assumed to be of equal length. The raw paper roll has a width B_{max}. In general, it is not possible to cut out an entire order without throwing away some of the raw paper. The optimum cutting scheme minimizes the waste paper or *trim loss*. In order to identify the best overall scheme, a maximum number of different cutting patterns P is postulated, where a pattern is defined

by the position of the knives. Each cutting pattern may have to be repeated several times in the overall scheme to meet the demand. There are N different product sizes and n_i rolls of size b_i must be cut. The existence of each pattern is denoted by a binary variable z_j, $j = 1, \ldots, P$. The number of repeats of pattern j is denoted by the integer variable m_j. The number of products of size i in pattern j is given by the integer variable r_{ij}. A sample cut is shown in Figure 22.7 for

$$N = 3,\ n = \begin{pmatrix} 4 \\ 5 \\ 5 \end{pmatrix},\ m = \begin{pmatrix} 2 \\ 1 \\ 1 \end{pmatrix} \text{ and } r = \begin{pmatrix} 0 & 2 & 3 \\ 4 & 1 & 0 \\ 0 & 0 & 2 \end{pmatrix}.$$

Some additional constraints are imposed on the cutting patterns. For instance, each pattern must have a minimum total width of $B_{max} - \Delta$. The maximum number of knives is limited to Nk_{max}. The general problem formulation is mostly linear except for the product-order constraints which are bilinear in the m_j and r_{ij} variables.

$$\min_{m_j, y_j, r_{ij}} \sum_{j=1}^{P} c_j m_j + C_j y_j$$

$$\begin{aligned}
s.t.\quad & \sum_j m_j\, r_{ij} \geq n_i,\ i = 1, \ldots, N \\
& (B_{max} - \Delta)\, y_j \leq \sum_i b_i\, r_{ij} \leq B_{max}\, y_j,\ j = 1, \ldots, P \\
& y_j \leq \sum_i r_{ij} \leq N k_{max}\, y_j,\ j = 1, \ldots, P \\
& y_j \leq m_j \leq M_j y_j,\ j = 1, \ldots, P \\
& \sum_j m_j \geq \max\left\{ \left\lceil \frac{\sum_i n_i}{N k_{max}} \right\rceil, \left\lceil \frac{\sum_i b_i n_i}{B_{max}} \right\rceil \right\} \\
& y_{k+1} \leq y_k,\ k = 1, \ldots, P-1 \\
& m_{k+1} \leq m_k,\ k = 1, \ldots, P-1 \\
& y_j \in \{0, 1\},\ j = 1, \ldots, P \\
& m_j \in [0, M_j] \cap \mathbb{N},\ j = 1, \ldots, P \\
& r_{ij} \in [0, N k_{max}] \cap \mathbb{N},\ i = 1, \ldots, N,\ j = 1, \ldots, P
\end{aligned}$$

where $c_j = 1$, $j = 1, \ldots, P$ and $C_j = 0.1\, j$, $j = 1, \ldots, P$.

This model is used for four problems and their explicit formulations can be found in Floudas *et al.* (1999), and Adjiman *et al.* (1998d). The number of

combinations of the integer variables are respectively 10^{16}, 10^{16}, 10^{23} and 10^{31}. The set of *feasible* combinations is smaller, but cannot be determined *a priori*. The value of the objective function at the global optimum solution is shown in Table 22.6 for each problem, together with values of the variables. There are usually several globally optimum sets of patterns and only one is listed here. An interesting feature of these problems is the small size of the integrality gap (Table 22.6). It indicates that the generation of tight lower bounds on the nonconvex MINLPs should not be a bottleneck.

This set of trim loss minimization problems is solved using the GMIN-αBB algorithm with different options. The nature of the problem is such that the lower bound often reaches the value of the global optimum solution even though not all variables are integer. Since convergence can only be declared when an integer solution is found, the performance statistics can vary dramatically from run to run. Each problem is therefore solved one hundred times and the mean and standard deviation of the number of iterations and CPU time are reported here.

Branching Variable Selection. When the branching variable is selected based on the most fractional variable at the solution of the lower bounding problem, all runs take over one hour to converge. Since the non-existence of a pattern i ($y_i = 0$) eliminates the corresponding m_i and r_{ij} variables from the problem, the y variables are assigned the highest priority. The m variables are branched on next, and the r variables follow. The results using this branching strategy, a maximum of one hundred αBB iterations and no bound updates are shown in Table 22.7. Although the average CPU time is greatly improved, the standard deviation is large.

Variable Bound Updates. The bounds on all integer variables are tightened through the optimization-based procedure. The results, presented in Table 22.8, show that the mean number of nodes decreases by 53 to 93% and the mean CPU time decreases by 74 to 94%.

Automatic αBB Termination. The test of lower bound improvement rate is imposed on the αBB runs using values of m and r that were successful for the pump network synthesis example, namely $m = 3$ iterations and $r = 30\%$. The results, shown in Table 22.9, indicate that the criterion used is too stringent for this problem as a few more nodes must be explored on average before the global optimum solution is found than when one hundred αBB iterations are allowed. However, the computational requirements are sufficiently close to the best values obtained so far to warrant adopting the automatic stopping criterion with $m = 3$ iterations and $r = 0.3$.

Summary

In this chapter, we discussed the theoretical and algorithmic issues of the GMIN-αBB global optimization approach, along with a variety of computa-

tional studies. The GMIN-αBB algorithm is designed to locate the global optimum solution of problems involving integer and continuous variables in functions whose continuous relaxation is twice continuously differentiable. The algorithm is similar to traditional branch-and-bound algorithms for mixed-integer problems in that branching occurs on the integer variables only and a continuous relaxation of the problem is constructed during the bounding step. It uses the αBB algorithm for the efficient and rigorous generation of lower bounds. The generality of the classes of mathematical problems that the GMIN-αBB can address, as well as its excellent performance on the literature problems, the pump network configuration problem and the four instances of the trim loss minimization problems make the GMIN-αBB a powerful global optimization approach. Additional analysis and discussion on the computational studies can be found in Adjiman *et al.* (1998d). Explicit mathematical formulations of the examples considered and complete solutions can be found in the recent book on test problems for local and global optimization by Floudas *et al.* (1999).

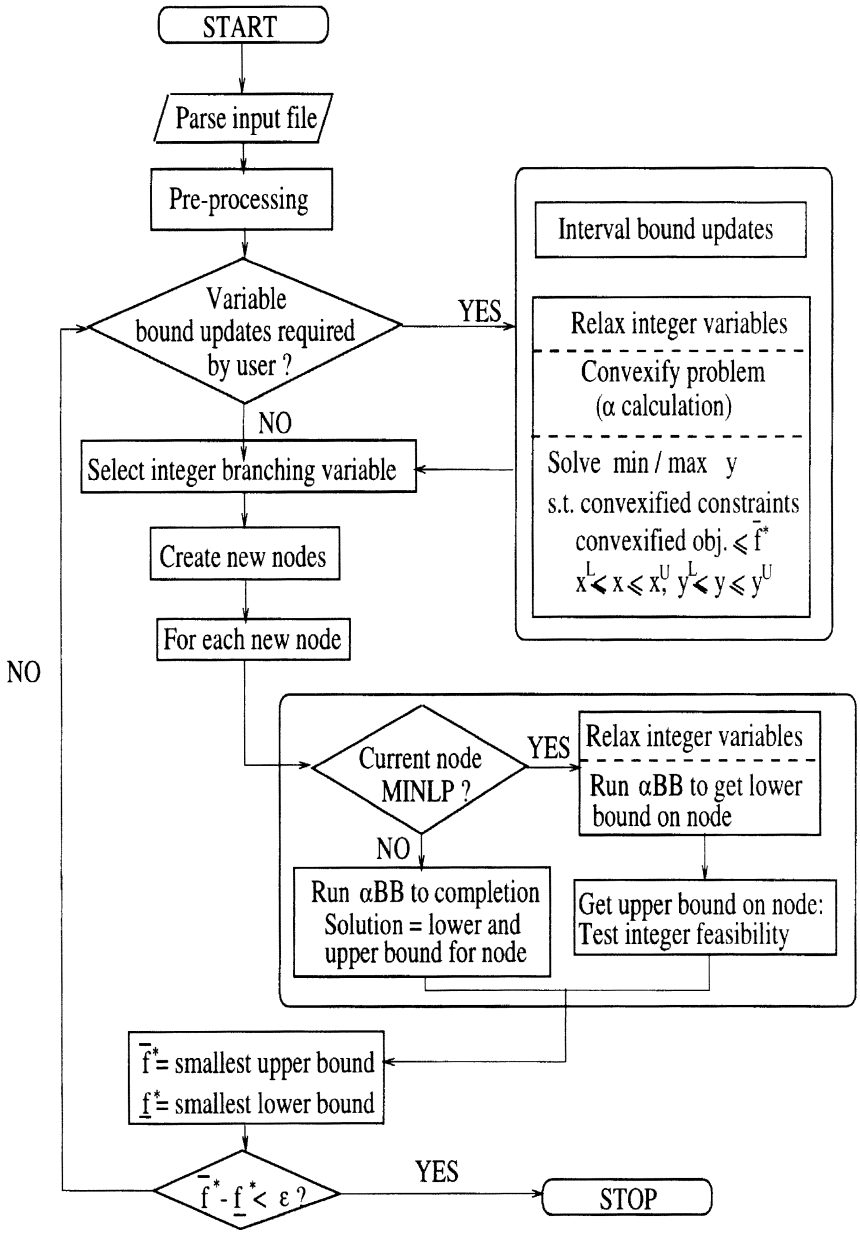


Figure 22.3. Flowchart for the GMIN-αBB algorithm

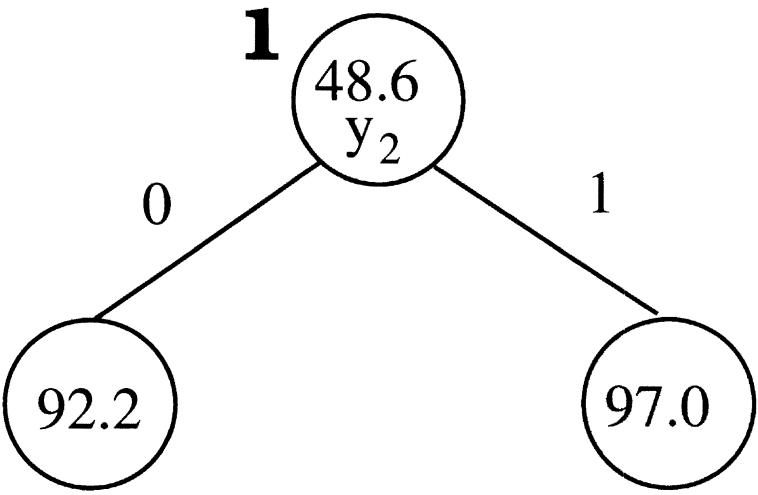


Figure 22.4. GMIN-αBB branch-and-bound tree

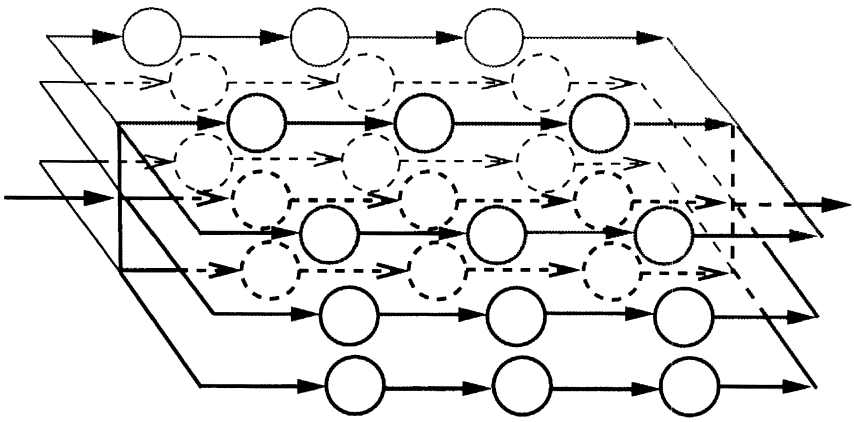


Figure 22.5. Superstructure for the pump network synthesis problem

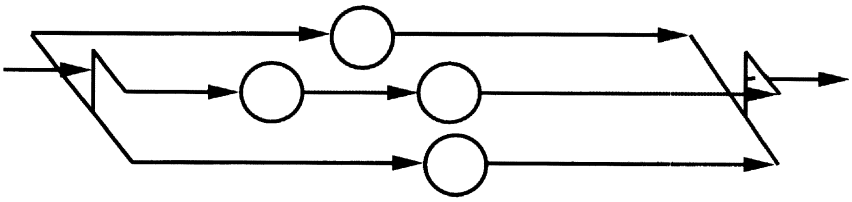


Figure 22.6. Global optimal configuration for pump network synthesis problem

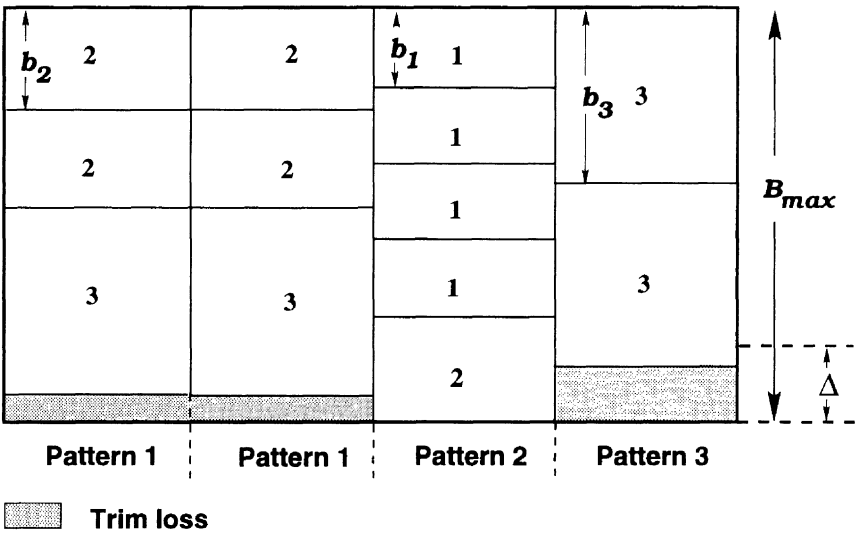


Figure 22.7. Trim loss minimization problem.

	Objective function	y	m	r	Integrality gap
Problem 1	19.6	$\begin{pmatrix}1\\1\\1\\0\end{pmatrix}$	$\begin{pmatrix}1\\3\\2\\0\end{pmatrix}$	$\begin{pmatrix}1&0&1&0\\2&0&0&0\\0&5&3&0\\2&0&1&0\end{pmatrix}$	0.50
Problem 2	8.6	$\begin{pmatrix}1\\0\\0\\0\end{pmatrix}$	$\begin{pmatrix}11\\0\\0\\0\end{pmatrix}$	$\begin{pmatrix}1&0&0&0\\1&0&0&0\\2&0&0&0\\1&0&0&0\end{pmatrix}$	0.50
Problem 3	10.3	$\begin{pmatrix}1\\0\\0\\0\\0\end{pmatrix}$	$\begin{pmatrix}15\\0\\0\\0\\0\end{pmatrix}$	$\begin{pmatrix}1&0&0&0&0\\1&0&0&0&0\\1&0&0&0&0\\1&0&0&0&0\\1&0&0&0&0\end{pmatrix}$	0.20
Problem 4	15.3	$\begin{pmatrix}1\\1\\0\\0\\0\\0\end{pmatrix}$	$\begin{pmatrix}8\\7\\0\\0\\0\\0\end{pmatrix}$	$\begin{pmatrix}1&0&0&0&0&0\\2&0&0&0&0&0\\0&2&0&0&0&0\\0&1&0&0&0&0\\0&2&0&0&0&0\\1&0&0&0&0&0\end{pmatrix}$	0.08

Table 22.6. Optimal solutions for four instances of the trim loss problem.

	GMIN-αBB nodes		CPU seconds	
	Mean	Std. dev.	Mean	Std. dev.
Problem 1	545	289	239	81
Problem 2	949	207	314	94
Problem 3	171	79	226	35
Problem 4	343	165	570	143

Table 22.7. Results for the trim loss minimization problems, with prioritized branching.

	GMIN-αBB nodes		CPU seconds	
	Mean	Std. dev.	Mean	Std. dev.
Problem 1	89	22	62	14
Problem 2	85	38	48	16
Problem 3	27	6	25	4
Problem 4	22	8	33	8

Table 22.8. Results for the trim loss minimization problems, with prioritized branching and optimization-based bound updates.

	GMIN-αBB nodes		CPU seconds	
	Mean	Std. dev.	Mean	Std. dev.
Problem 1	95	30	60	18
Problem 2	105	37	55	19
Problem 3	37	6	28	4
Problem 4	44	11	54	12

Table 22.9. Results for the trim loss minimization problems, with prioritized branching, optimization-based bound updates and automatic αBB run termination.

V Nonlinear Constrained Systems of Equations

23 ALL SOLUTIONS OF NONLINEAR CONSTRAINED SYSTEMS OF EQUATIONS

In this chapter, we discuss the enclosure of all solutions of nonlinear constrained systems of equations. Section 23.1 provides the motivation from the applications' point of view and reviews briefly the major previous approaches. Section 23.2 defines the problem and presents the key elements of the global optimization approach. Section 23.3 discusses several theoretical results for the convex lower bounding problems. Section 23.4 presents the global optimization based algorithmic approach for enclosing all solutions of constrained systems of equations. Finally, section 23.5 presents several computational studies on problems taken from the literature. The material presented in this chapter is based on the work of Maranas and Floudas (1995).

23.1 INTRODUCTION

The enclosure of all solutions of nonlinear constrained systems of equations is regarded as a very challenging task in applied mathematics, engineering and sciences. Representative applications include (i) the identification of potential multiplicities of exothermic reactions occurring in adiabatic CSTR's; (ii) the solution of cubic equations in PVT calculations for the vapor, liquid and metastable state specific volumes; (iii) evaluation of multiple–steady states in certain types of reaction networks; (iv) the identification of multiple solutions for porous catalysts effectiveness factors in exothermic reactions; (v) the solution of (un)constrained simulation problems in steady–state flowsheet mod-

C. A. Floudas, *Deterministic Global Optimization*
© Springer Science+Business Media Dordrecht 2000

elling; (vi) the identification of equilibrium points in (un)reacting multiphase systems such as multiple miscibility gaps in liquid–liquid phase equilibria; (vii) the prediction of all homogeneous, heterogeneous and reactive azeotropes, as well as eutectic points at some temperature or pressure for given activity coefficient models; (viii) the location of all multiple steady states of individual distillation columns and of interlinked separation systems; (ix) the prediction of multiple steady–states in various complex reactor systems; and (x) the prediction of multiplicities occurring in fluid mechanics when solving nonlinear boundary value problems.

There exists a large body of literature on methods for solving systems of equations. These approaches fall within three broad classes: (i) Newton and quasi–Newton type methods; (ii) homotopy continuation type (embedding) methods; (iii) Interval–Newton methods; and (iv) deterministic global optimization methods.

Newton and quasi–Newton type methods and their modifications achieve superlinear convergence only when they are well within the neighborhood of the solution. However, these methods are likely to fail if the initial guess is poor or if singular points are encountered. Modifications in an attempt to avoid singularities may incorporate trust–region techniques such as Powell's "dogleg" method (Powell, 1970), steepest descent direction information as in Chen and Stadtherr (1984), Duff *et al.* (1987), Moré (1987) and alterations on the quasi–Newton Jacobian estimates by Paloschi and Perkins (1988). These methods, although very computationally efficient, cannot provide robust guarantees for convergence.

The second class of methods are the *embedding methods.* These are known as *continuation, homotopy continuation, or incremental loading* and are based on the pioneering work of Lahaye (1934), Leray and Schauder (1934), Davidenko (1953), and Klopfenstein (1961). A problem common to all homotopy variants is that variable bounds and inequality constraints cannot be handled directly. A comprehensive review of the extensive literature in this area can be found in Garcia and Zangwill (1981). Application of the homotopy ideas for a variety of chemical engineering problems has been very extensive (Salgovic *et al.*, 1981; Bhargava and Hlavacek, 1984; Wayburn and Seider, 1983; Chavez *et al.*, 1986; Hlavacek, 1985; Byrne and Baird, 1985; Bekiaris *et al.*, 1993). While in practice homotopy continuation methods are frequently used in an attempt to locate all solutions of arbitrary nonlinear systems of equations, mathematical guarantees that all solutions will be found, exist only in special cases (e.g., polynomial systems with no constraints). For polynomial systems of equations, however, Morgan (1987) proposed a differential arclength continuation using a special homotopy that establishes a number of continuation paths guaranteed to converge to all possible real and complex roots. Two popular software packages, CONSOL by Morgan (1987) and POLSYS by Watson *et al.* (1987) have implemented this method.

Interval–Newton methods can enclose all solutions of nonlinear systems of equations lying within certain variable bounds. They do so by applying the

classical Newton-like iterative methods on interval variables coupled with a generalized bisection strategy. Interval computations are discussed in detail in Neumaier (1990) and Hansen (1992). A version of the basic Interval–Newton method has been implemented into the program INTBIS by Kearfott and Novoa (1990) which is coupled with a portable interval standard function library INTLIB by Kearfott *et al.* (1994). By utilizing these routines, Seader and Kumar (1994) studied the presence of multiple steady–states in certain process simulation examples. Schnepper and Stadtherr (1996) developed a parallel implementation to address a number of process flowsheeting problems. McKinnon and Mongeau (1998), Hua *et al.* (1996), and Hua *et al.* (1998), applied interval analysis methods for the phase and chemical reaction equilibrium problems, while Maier *et al.* (1998) studied the enclosure of all homogeneous azeotropes based on the formulations proposed by Harding *et al.* (1997). The main attractive feature of Interval–Newton methods is that they enclose all solutions of fairly arbitrary nonlinear systems of equations within certain variable bounds. However, this wide applicability to almost arbitrary nonlinear functions comes at an expense. Because no specific structure of the individual expressions participating in the nonlinear equations is analyzed, the obtained lower/upper bounds on these nonlinear expressions tend to be fairly loose.

Deterministic global optimization approaches for enclosing all solutions of nonlinear constrained systems of equations, which are the focus of this chapter, are based on a branch and bound framework which explores the entire space and therefore they can provide guarantees for enclosing all solutions. The fundamental difference between the deterministic global optimization approaches and Interval–Newton methods is that while the latter bounds functions with scalars, the former exploit the mathematical structure of the problem, so as to underestimate the nonconvex functions by tight convex functions resulting in improved bounds.

23.2 PROBLEM DEFINITION

The problem of enclosing all solutions of a nonlinear system of equations subject to inequality constraints and variable bounds is formulated as:

$$\begin{aligned} h_j(\boldsymbol{x}) &= 0, \ j \in \mathcal{N}_E \\ g_k(\boldsymbol{x}) &\leq 0, \ k \in \mathcal{N}_I \\ \boldsymbol{x}^L &\leq \boldsymbol{x} \leq \boldsymbol{x}^U, \end{aligned} \tag{23.1}$$

where $\mathcal{N}_E$ is the set of equalities, $\mathcal{N}_I$ the set of inequality constraints, and $\boldsymbol{x}$ the vector of variables.

Remark 23.2.1 Note that in formulation (23.1), the total number of variables is allowed to be different from the total number of equalities so that neither the existence nor the uniqueness of a solution of (23.1) is postulated. Therefore,

overspecified and underspecified systems are included in the present investigations. A number of important problems naturally arise as special instances of formulation (23.1). Eliminating all inequality constraints, (23.1) corresponds to a system of nonlinear equations. Eliminating all equality constraints (23.1) checks the existence of feasible points for the given inequality constraint set.

Formulation (23.1) can be transformed into the following min–max optimization problem (Horst and Tuy, 1990) :

$$\min_{\boldsymbol{x}} \max_{j \in \mathcal{N}_E} \quad |h_j(\boldsymbol{x})|$$

$$\text{subject to} \quad g_k(\boldsymbol{x}) \leq 0, \; k \in \mathcal{N}_I$$

$$\boldsymbol{x}^L \leq \boldsymbol{x} \leq \boldsymbol{x}^U.$$

Introducing a single slack variable s, the min-max problem can be written as the following optimization problem (23.2):

$$\begin{aligned} \min_{\boldsymbol{x}, s \geq 0} \quad & s \\ \text{s.t.} \quad h_j(\boldsymbol{x}) - s &\leq 0, \; j \in \mathcal{N}_E \\ -h_j(\boldsymbol{x}) + s &\leq 0, \; j \in \mathcal{N}_E \\ g_k(\boldsymbol{x}) &\leq 0, \; k \in \mathcal{N}_I \\ \boldsymbol{x}^L \leq \boldsymbol{x} &\leq \boldsymbol{x}^U. \end{aligned} \tag{23.2}$$

Remark 23.2.2 Notice that there is a one to one correspondence between multiple global minima $(\boldsymbol{x}^*, s^*)$ of (23.2) for which $s^* = 0$ and solutions of (23.1). This means that if the global minimum of (23.2) involves a nonzero slack variable s^* then the original problem (23.1) has no solutions. Note that, unless the functions $h_j(\boldsymbol{x})$ are *linear* and $g_k(\boldsymbol{x})$ are *convex*, formulation (23.3) corresponds to a *nonconvex* optimization problem. This implies that if a local optimization approach is used to solve (23.2) one might miss some of the multiple global minima of (23.2) or even erroneously deduce that there are no solutions for (23.1). Therefore, an approach that is guaranteed to always enclose *all* multiple global minima of (23.2) appears to be necessary for solving (23.1) so as (i) the correct solution vector $(\boldsymbol{x}^*, s^*)$ is identified and (ii) all solutions $(\boldsymbol{x}^*)$ of (23.1) with $s^* = 0$ are found.

In this chapter, we present a deterministic global optimization based on the work of Maranas and Floudas (1995). This is guaranteed to enclose all ϵ–global minima of (23.2) through the successive refinement of converging lower and upper bounds on the solution based on the solution of convex optimization problems defined by a branch and bound framework. A lower bound on the

solution of (23.2) is found by first replacing each nonconvex constraint in (23.2) with a convex underestimation of it and then subsequently solving the convex relaxation of (23.2). This approach naturally partitions the constraints of formulation (23.2) into convex constraints for which no relaxation is required and nonconvex ones. This partitioning yields the following alternative formulation (23.3):

$$\begin{aligned} \min_{\boldsymbol{x}, s \geq 0} \quad & s \\ \text{s.t.} \quad h_j^{nonc}(\boldsymbol{x}) - s &\leq 0, \; j \in \mathcal{N}_{noncE} \\ -h_j^{nonc}(\boldsymbol{x}) + s &\leq 0, \; j \in \mathcal{N}_{noncE} \\ g_k^{nonc}(\boldsymbol{x}) &\leq 0, \; k \in \mathcal{N}_{noncI} \\ h_j^{lin}(\boldsymbol{x}) &= 0, \; j \in \mathcal{N}_{linE} \\ g_k^{conv}(\boldsymbol{x}) &\leq 0, \; j \in \mathcal{N}_{convI} \\ \boldsymbol{x}^L \leq \boldsymbol{x} &\leq \boldsymbol{x}^U, \end{aligned} \tag{23.3}$$

where $\mathcal{N}_{noncE}, \mathcal{N}_{linE}$ are the sets of nonconvex and linear equality constraints respectively, and $\mathcal{N}_{noncI}, \mathcal{N}_{convE}$ are the sets of nonconvex and convex inequality constraints,

$$\mathcal{N}_E = \mathcal{N}_{noncE} \cup \mathcal{N}_{linE}, \quad \mathcal{N}_I = \mathcal{N}_{noncI} \cup \mathcal{N}_{convI}.$$

A convex relaxation (23.4) of (23.3) of the form:

$$\begin{aligned} \min_{\boldsymbol{x}, s \geq 0} \quad & s \\ \text{s.t.} \quad \hat{h}_{+,j}^{nonc}(\boldsymbol{x}) - s &\leq 0, \; j \in \mathcal{N}_{noncE} \\ \hat{h}_{-,j}^{nonc}(\boldsymbol{x}) + s &\leq 0, \; j \in \mathcal{N}_{noncE} \\ \hat{g}_k^{nconc}(\boldsymbol{x}) &\leq 0, \; k \in \mathcal{N}_{noncI} \\ h_j^{lin}(\boldsymbol{x}) &= 0, \; j \in \mathcal{N}_{linE} \\ g_k^{conv}(\boldsymbol{x}) &\leq 0, \; j \in \mathcal{N}_{convI} \\ \boldsymbol{x}^L \leq \boldsymbol{x} &\leq \boldsymbol{x}^U, \end{aligned} \tag{23.4}$$

can be obtained by replacing the original nonconvex functions

$$h_j^{nonc}(\boldsymbol{x}), \quad -h_j^{nonc}(\boldsymbol{x}), \quad g_k^{nonc}(\boldsymbol{x})$$

with some convex tight lower bounding functions

$$\hat{h}_{+,j}^{nonc}(\boldsymbol{x}), \quad \hat{h}_{-,j}^{nonc}(\boldsymbol{x}), \quad \hat{g}_k^{nonc}(\boldsymbol{x}).$$

These lower bounding functions $\hat{h}_{+,j}^{nonc}(\boldsymbol{x})$,$\hat{h}_{-,j}^{nonc}(\boldsymbol{x})$, $\hat{g}_k^{nonc}(\boldsymbol{x})$ must be (i) convex in $\left[\boldsymbol{x}^L, \boldsymbol{x}^U\right]$, (ii) valid underestimators of the original functions h_j^{nonc}, $-h_j^{nonc}$, g_k^{nonc} respectively, and (iii) for every point $\boldsymbol{x} \in \left[\boldsymbol{x}^L, \boldsymbol{x}^U\right]$ the maximum separation between the original functions and the convex underestimators must become arbitrarily ϵ small by appropriately reducing the size of the rectangular domain $\left[\boldsymbol{x}^L, \boldsymbol{x}^U\right]$ around the point $\boldsymbol{x}$ inside which the convex underestimators are defined. These requirements are expressed mathematically as follows:

Property 23.2.1 $\hat{h}_{+,j}^{nonc}(\boldsymbol{x})$, $\hat{h}_{-,j}^{nonc}(\boldsymbol{x})$, and $\hat{g}_k^{nonc}(\boldsymbol{x})$ be convex $\forall\, \boldsymbol{x} \in \left[\boldsymbol{x}^L, \boldsymbol{x}^U\right]$.

Property 23.2.2 $h_j^{nonc}(\boldsymbol{x}) \geq \hat{h}_{+,j}^{nonc}(\boldsymbol{x})$, $-h_j^{nonc}(\boldsymbol{x}) \geq \hat{h}_{-,j}^{nonc}(\boldsymbol{x})$, and $g_k^{nonc}(\boldsymbol{x}) \geq \hat{g}_k^{nonc}(\boldsymbol{x})$, $\forall\, \boldsymbol{x} \in \left[\boldsymbol{x}^L, \boldsymbol{x}^U\right]$.

Property 23.2.3 $\forall\, \boldsymbol{x} \in \left[\boldsymbol{x}^L, \boldsymbol{x}^U\right]$ and $\epsilon > 0$, $\exists\, \left[\boldsymbol{x}^L, \boldsymbol{x}^U\right] \subseteq \left[\boldsymbol{x}^L, \boldsymbol{x}^U\right]$ with $\delta(\epsilon) = \|\boldsymbol{x}^U - \boldsymbol{x}^L\|_2^{1/2} > 0$ such that

$$\max_{\boldsymbol{x}\in[\boldsymbol{x}^L,\boldsymbol{x}^U]} \left(h_j^{nonc}(\boldsymbol{x}) - \hat{h}_{+,j}^{nonc}(\boldsymbol{x})\right) < \epsilon,$$

$$\max_{\boldsymbol{x}\in[\boldsymbol{x}^L,\boldsymbol{x}^U]} \left(-h_j^{nonc}(\boldsymbol{x}) - \hat{h}_{-,j}^{nonc}(\boldsymbol{x})\right) < \epsilon,$$

$$\max_{\boldsymbol{x}\in[\boldsymbol{x}^L,\boldsymbol{x}^U]} \left(g_k^{nonc}(\boldsymbol{x}) - \hat{g}_k^{nonc}(\boldsymbol{x})\right) < \epsilon.$$

Remark 23.2.3 Property 23.2.3 requires that the maximum separation between the nonconvex function and its tight convex lower bounding function, defined inside some rectangular region, must go to zero

$$\lim_{\delta\to 0^+} \epsilon = 0.$$

as the size of the rectangular domain approaches zero ($\delta = 0$). This is important for proving finite ϵ-convergence. The order $\mathcal{O}(\epsilon) = \mathcal{O}(\delta^n)$ with which ϵ approaches zero determines the speed of convergence. The largest possible value for n is desirable so that the maximum tolerance reaches an arbitrary value ϵ for a not too small variable range δ. For example, if the maximum separation ϵ goes as δ^2 then a value of just $\delta = 0.01$ suffices to meet a convergence tolerance of $\epsilon = 0.0001$.

23.3 CONVEX LOWER BOUNDING

An efficient convex lower bounding of the nonconvex functions appearing in formulation (23.3) is central to the design of the deterministic global optimization approach for enclosing all solutions of nonlinear constrained systems of equations. As we discussed in section 2.4 of Chapter 2, the tightest possible

convex lower bounding function for any arbitrary nonconvex function $f(\boldsymbol{x})$ inside some rectangular region S is called the *convex envelope* $\phi(\boldsymbol{x})$ of $f(\boldsymbol{x})$. The convex envelope satisfies the following properties :

(i) $\phi(\boldsymbol{x})$ is convex for all $\boldsymbol{x} \in S$.

(ii) $f(\boldsymbol{x}) \geq \phi(\boldsymbol{x})$ for all $\boldsymbol{x} \in S$.

(iii) For all functions $g(\boldsymbol{x})$ that satisfy (i) and (ii), $\phi(\boldsymbol{x}) \geq g(\boldsymbol{x})$ for all $\boldsymbol{x} \in S$.

There exist several techniques for obtaining functions that satisfy properties (i) and (ii). A number of convex lower bounding procedures are discussed next.

23.3.1 Products of univariate functions

Products of univariate functions f_i,

$$f(\boldsymbol{x}) = \prod_{i=1}^{N} f_i(x_i)$$

are in general nonconvex functions even if the corresponding univariate functions are convex. By utilizing appropriate linear transformations a large number of nonlinearities appearing in chemical engineering problems can be modeled through the use of products of univariate functions. For instance, in mass and energy balance equations, bilinear or even trilinear terms such as products between mole fractions, concentrations, flowrates and temperatures

$$x_A T, \;\; C_A C_B, \;\; x_i C, \;\; F T x_i, \;\; F T, \;\; C T x$$

are the primary source of nonlinearities. Specifically, in phase equilibria, terms involving products of mole fractions with logarithms and ratios of linear terms, appear in different activity models:

(i) $\sum_j \frac{x_j \Lambda_{ji}}{\sum_k x_k \Lambda_{jk}}$ $(Wilson)$,

(ii) $\sum_j \frac{\tau_{ji} G_{ji} x_j}{\sum_j G_{ji} x_j}$ $(NRTL)$.

In physical properties estimation, we have polynomial terms in PVT equations of state:

(i) $V^3 - [RT/p]\, V^2 + \left[a\alpha - bRT - pb^2\right] V - (a\alpha b/p)$ (SRK equation of state).

In reaction engineering, we have products and fractions of compositions in complex kinetics, products between exponential factors and compositions, and fractionals in CSTR design equations:

(i) $\frac{k_1 C_A}{1+KC_B}$ (Complex kinetics),

(ii) $k_0 \exp\left(-\frac{E}{RT}\right) C_A$ (Nonisothermal reaction),

(iii) $\frac{\theta A \exp(-E/RT)}{1+\theta A \exp(-E/RT)}$ (nonisothermal CSTR).

All the aforementioned terms can be decomposed into the product of two or more univariate functions.

As we discussed in section 2.4.4 of Chapter 2, the convex envelope of a nonconvex bilinear product of xy inside the rectangular domain $\left[x^L, x^U\right] \times \left[y^L, y^U\right]$ takes the form :

$$\max\left(x^L y + x y^L - x^L y^L,\ x^U y + x y^U - x^U y^U\right),$$

while the maximum separation between the bilinear term xy and its convex envelope is equal to one fourth of the area of the rectangular domain $\left[x^L, x^U\right] \times \left[y^L, y^U\right]$.

As we discussed in section 2.4.5 of Chapter 2, the conditions under which a lower bounding function can be obtained for the product of two arbitrary univariate functions $f(x)$ and $g(y)$ are :

Theorem 23.1 *If f, g are twice differentiable univariate functions $f(x), g(y) \in C^2$ defined inside a rectangle $\left[\left(x^L, x^U\right), \left(y^L, y^U\right)\right]$ and*

$$\begin{aligned} l(x,y) \;=\; \max\ \{ \quad & \phi\left(f^L g(y)\right) + \phi\left(g^L f(x)\right) - f^L g^L, \\ & \phi\left(f^U g(y)\right) + \phi\left(g^U f(x)\right) - f^U g^U \} \end{aligned}$$

$$\begin{aligned} \textit{where } f^L &= \inf_{x^L \le x \le x^U} f(x), \\ f^U &= \sup_{x^L \le x \le x^U} f(x), \\ g^L &= \inf_{x^L \le x \le x^U} g(x), \\ g^U &= \sup_{x^L \le x \le x^U} g(x) \end{aligned}$$

and $\phi\left(f^L g(y)\right), \phi\left(g^L f(x)\right), \phi\left(f^U g(y)\right)$, and $\phi\left(g^U f(x)\right)$ are the convex envelopes of the univariate functions $f^L g(y), g^L f(x), f^U g(y)$, and $g^U f(x)$ respectively then:

(i) $l(x,y)$ is convex, $\forall\,(x,y) \in \left[x^L, x^U\right] \times \left[y^L, y^U\right]$.

(ii) $f(x)g(y) \ge l(x,y)$, $\forall\,(x,y) \in \left[x^L, x^U\right] \times \left[y^L, y^U\right]$.

Proof. See Chapter 2.

Also, as stated in section 2.4.5 of Chapter 2, under certain conditions $l(x,y)$ corresponds to the actual convex envelope of the product $f(x)g(y)$. These conditions are :

Theorem 23.2 *If (i) the univariate functions,*

$$g^L f(x),\ g^U f(x) \textit{ and } f^L g(y),\ f^U g(y)$$

are concave in $[x^L, x^U]$ *and* $[y^L, y^U]$ *respectively and (ii) the functions* $f(x), g(y)$ *are monotonic, then* $l(x,y)$ *is the convex envelope of* $f(x)g(y)$.

$$\phi(f(x)g(y)) = l(x,y), \ \forall\,(x,y) \in [x^L, x^U] \times [y^L, y^U].$$

Proof. See Chapter 2.

The following theorem extends the convex lower bounding results to the product of N univariate functions.

Theorem 23.3 *If* $f_i \in \mathcal{C}^2 : [x_i^L, x_i^U] \to \mathcal{R}$, $i = 1, \ldots, N$ *and*

$$\begin{aligned}
L(\mathbf{x}) &= y_0 \\
\textit{where} \quad y_j &= \max\{\phi(f_{j+1}^L y_{j+1}) + \phi(y_{j+1}^L f_{j+1}(x_{j+1})) - y_{j+1}^L f_{j+1}^L, \\
&\qquad \phi(f_{j+1}^U y_{j+1}) + \phi(y_{j+1}^U f_{j+1}(x_{j+1})) - y_{j+1}^U f_{j+1}^U, \}, \\
&\qquad j = 0, \ldots, N-3 \\
y_{N-2} &= \max\{\phi(f_{N-1}^L f_N(x_N)) + \phi(f_N^L f_{N-1}(x_{N-1})) - f_{N-1}^L f_N^L, \\
&\qquad \phi(f_{N-1}^U f_N(x_N)) + \phi(f_N^U f_{N-1}(x_{N-1})) - f_{N-1}^U f_N^U\} \\
\textit{and} \quad y_j^{L/U} &= \inf_{x_{j+1}, y_{j+1}} /\sup \{\phi(f_{j+1}^L y_{j+1}) + \phi(y_{j+1}^L f_{j+1}(x_{j+1})) - y_{j+1}^L f_{j+1}^L, \\
&\qquad \phi(f_{j+1}^U y_{j+1}) + \phi(y_{j+1}^U f_{j+1}(x_{j+1})) - y_{j+1}^U f_{j+1}^U, \}, \\
&\qquad j = 0, \ldots, N-3 \\
y_{N-2}^{L/U} &= \inf_{x_N, x_{N-1}} /\sup \{\phi(f_{N-1}^L f_N(x_N)) + \phi(f_N^L f_{N-1}(x_{N-1})) - f_{N-1}^L f_N^L, \\
&\qquad \phi(f_{N-1}^U f_N(x_N)) + \phi(f_N^U f_{N-1}(x_{N-1})) - f_{N-1}^U f_N^U\}
\end{aligned}$$

then

(i) $L(\boldsymbol{x})$ *is convex,* $\forall\, \boldsymbol{x} \in [\boldsymbol{x}^L, \boldsymbol{x}^U]$.

(ii) $\prod_{i=1}^{N} f_i(x_i) \geq L(\boldsymbol{x}), \ \forall\, \boldsymbol{x} \in [\boldsymbol{x}^L, \boldsymbol{x}^U]$.

Proof. Starting from the beginning of the recursive definition of $L(\boldsymbol{x})$, y_{N-2} is a convex function of (x_{N-1}, x_N) as the max of two convex functions. For the same reason y_{N-3} is a convex function of (x_{N-2}, y_{N-2}) or otherwise of (x_{N-2}, x_{N-1}, x_N). By recursively substituting y_j into the expression for y_{j-1} we deduce that for every $j = 0, \ldots, N-2$, y_j is a convex function. Therefore, $L(\boldsymbol{x}) = y_0$ is a convex function of $(x_1, x_2, \ldots, x_N)$ which proves part (i) of Theorem 23.3.

From Theorem 23.1 and the statement of Theorem 23.3 we have,

$$\begin{aligned} f_{N-1}(x_{N-1})f_N(x_N) &\geq \max\{\phi(f_{N-1}^L f_N(x_N)) + \phi(f_N^L f_{N-1}(x_{N-1})) - f_{N-1}^L f_N^L, \\ &\quad \phi(f_{N-1}^U f_N(x_N)) + \phi(f_N^U f_{N-1}(x_{N-1})) - f_{N-1}^U f_N^U\} \\ &= y_{N-2} \end{aligned}$$

$$\begin{aligned} \text{and} \quad f_{j+1}(x_{j+1})y_{j+1} &\geq \max\{\phi(f_{j+1}^L y_{j+1}) + \phi\left(y_{j+1}^L f_{j+1}(x_{j+1})\right) - y_{j+1}^L f_{j+1}^L, \\ &\quad \phi(f_{j+1}^U y_{j+1}) + \phi\left(y_{j+1}^U f_{j+1}(x_{j+1})\right) - y_{j+1}^U f_{j+1}^U,\} \\ &= y_j, \quad j = 0, \ldots, N-3 \end{aligned}$$

By combining these last two sets of inequalities we have,

$$\prod_{i=1}^{N} f_i(x_i) \geq y_0 = L(\boldsymbol{x}), \ \forall\, \boldsymbol{x} \in [\boldsymbol{x}^L, \boldsymbol{x}^U]$$

which proves part (ii) of Theorem 23.3.

Theorem 23.3 describes one possible sequence of recursively combining pairs of univariate functions. Theorem 23.4 states how many of these alternative sequences exist for convex lower bounding the product of N univariate functions.

Theorem 23.4 *There are,*

$$\frac{(N!)^2}{N2^{N-1}},$$

ways of combining pairs of univariate functions which yield different convex lower bounding schemes for a product of N univariate functions.

Proof. Clearly, there exist $\binom{N}{2} = \frac{N(N-1)}{2}$ ways of selecting the first pair of univariate functions to be convex lower bound. After this action, we are left with $N-1$ functions which implies that there are $\binom{N-1}{2} = \frac{(N-1)(N-2)}{2}$ alternatives for picking the next pair of functions. This recursive convex lower bounding is continued until we are left with only a pair of functions involving a single convex lower bounding alternative. Because every convex lower bounding stage is independent of the previous one, the total number of ways of combining N univariate functions in pairs of two is:

$$\prod_{i=2}^{N} \binom{i}{2} = \prod_{i=2}^{N} \frac{i\,(i-1)}{2} = \frac{N!(N-1)!}{2^{N-1}} = \frac{(N!)^2}{N2^{N-1}}.$$

Maranas and Floudas (1995) proposed necessary and sufficient conditions for convexity/concavity of generalized polynomial expressions

$$\left(\prod_{i=1}^{N} x_i^{d_i}\right), \quad d_i \in \Re, \ i = 1, \ldots, N$$

which are a special case of products of univariate functions so as to identify certain convexity and/or concavity functions. These are reflected in the following three theorems.

Theorem 23.5 *If one of the following conditions holds,*

(1) $x, y \geq 0$.

(2) a, b *are even integers.*

(3) a, b *are odd integers, and* $xy \geq 0$.

(4) a, b *are integers,* a *is odd,* b *is even, and* $x \geq 0$.

(5) a, b *are integers,* a *is even,* b *is odd, and* $y \geq 0$

then (a) $f = x^a y^b$ *is convex in* (x, y) *if one of the following is true:*

(i) $a \leq 0,\ b \leq 0$.

(ii) $a \leq 0,\ 1 - a - b \leq 0$.

(iii) $b \leq 0,\ 1 - a - b \leq 0$.

and (b) $f = x^a y^b$ *is concave in* (x, y) *if*

(i) $a \geq 0,\ b \geq 0,\ a + b \leq 1$.

Proof. See Maranas and Floudas (1995).

Theorem 23.6 *If one of the following conditions holds,*

(1) a, b *are odd integers, and* $xy \leq 0$.

(2) a, b *are integers,* a *is odd,* b *is even, and* $x \leq 0$.

(3) a, b *are integers,* a *is even,* b *is odd, and* $y \leq 0$

then (a) $f = x^a y^b$ *is concave in* (x, y) *if one of the following is true:*

(i) $a \leq 0,\ b \leq 0$.

(ii) $a \leq 0,\ 1 - a - b \leq 0$.

(iii) $b \leq 0,\ 1 - a - b \leq 0$.

and (b) $f = x^a y^b$ *is convex in* (x, y) *if*

(i) $a \geq 0,\ b \geq 0,\ a + b \leq 1$.

Proof. See Maranas and Floudas (1995).

A similar set of conditions for convexity/concavity can be obtained for the general n–dimensional case. For the sake of simplicity, we assume that $x_i \geq 0,\ i = 1, \ldots N$ which can always be achieved with a simple rescaling of variables.

Theorem 23.7 *The function* $f : \Re_+^N \to \Re_+$, $f(\mathbf{x}) = \prod_{i=1}^{N} x_i^{d_i}$ *(a) is convex in* $\mathbf{x} \in \Re_+^N$ *if one of the following conditions is true:*

(i) $d_i \leq 0, \forall\, i = 1, \ldots, N$

(ii) $\exists\, j$ *such that* $d_j \geq 1 - \sum_{i \neq j}^{N} d_i$, *.and* $d_i \leq 0, \forall\, i \neq j,\ i = 1, \ldots, N$

and (b) is concave in $\mathbf{x} \in \Re_+^N$ *if*

(iii) $d_i \geq 0, \forall\, i = 1, \ldots, N$, *and* $\sum_{i=1}^{N} d_i \leq 1$.

Proof. See Maranas and Floudas (1995).

From the analysis in this subsection it is clear that in order to derive convex lower bounds for the product of univariate functions it is important to obtain the convex envelope, or at least a tight convex lower bounding function, of arbitrary univariate functions. A systematic procedure for constructing the convex envelope of arbitrary functions of a single variable inside a certain interval is presented in section 2.4.6 of Chapter 2.

23.3.2 Convex Lower Bounding Using αBB

As we discussed in Chapters 11 and 12, for arbitrary nonconvex functions that are twice continuously differentiable $f \in \mathcal{C}^2 : \left[\boldsymbol{x}^L, \boldsymbol{x}^U\right] \to \mathcal{R}$, a convex lower bounding function $\mathcal{L}$ of f can be defined by augmenting f with the addition of a separable convex quadratic function of $\boldsymbol{x}$ as proposed in Maranas and Floudas (1994a) and generalized to include equality and inequality constraints in (Androulakis *et al.*, 1995).

$$\mathcal{L}(\boldsymbol{x}) = f(\boldsymbol{x}) + \alpha \left(\boldsymbol{x}^L - \boldsymbol{x}\right)^T \left(\boldsymbol{x}^U - \boldsymbol{x}\right)$$

$$\text{where} \qquad \alpha \geq \max\left\{0, -\frac{1}{2} \min_{\substack{k \\ \boldsymbol{x}^L \leq \boldsymbol{x} \leq \boldsymbol{x}^U}} \lambda_k(\boldsymbol{x})\right\}.$$

Note that α is a nonnegative parameter which must be greater than or equal to the negative one half of the minimum eigenvalue of f over $\boldsymbol{x}^L \leq \boldsymbol{x} \leq \boldsymbol{x}^U$. In Chapters 10 and 11 we discussed a variety of rigorous methods for the calculation of the α parameter. The effect of adding the extra separable quadratic term to f is to make $\mathcal{L}$ convex by overpowering the nonconvexity characteristics of f with the addition of the term 2α to the diagonal elements of its Hessian matrix. As we have discussed in Chapters 11 and 12 this function $\mathcal{L}$ defined over the rectangular domain $\left[\boldsymbol{x}^L, \boldsymbol{x}^U\right]$, involves a number of properties which enable us to use it as a tight convex lower bounding function of f.

23.4 PROCEDURE FOR ENCLOSING ALL SOLUTIONS

23.4.1 Description

A deterministic global optimization approach aims at enclosing all ϵ–solutions of nonlinear systems of equalities subject to nonlinear inequality constraints (23.1). By introducing a slack variable, the initial problem (23.1) is transformed into a global optimization problem (23.3) whose multiple global minima (if any) correspond to the multiple solutions of (23.1). A zero objective value denotes the existence of a solution whereas a strictly positive objective function value implies that (23.1) has no solutions. This defines an one–to–one correspondence between solutions of the constrained system of equations (23.1) and multiple global minima with an objective value of zero for problem (23.3).

Even if the number of multiple global minima of (23.3) is finite, Hansen *et al.* (1992b) have shown that no algorithm can locate all of them with a finite number of function evaluations. Furthermore, a corrolary of this result (Hansen *et al.*, 1992b) is that no algorithm can always localize, with a finite number of function evaluations, all globally optimal points by compact subintervals in one–to–one correspondence with them. Therefore, a more reasonable aim than finding all global minima of (23.3) is to find arbitrarily small disjoint subintervals containing all globally optimal points of (23.3) if any.

These multiple ϵ–global minima of (23.3), (if any) can then be localized based on a branch and bound procedure involving the successive refinement of convex relaxations (23.4) of the initial problem (23.3). Formulation (23.4) is obtained by replacing the nonconvex functions $h_j^{nonc}, -h_j^{nonc}, g_k^{nonc}$ with some *tight, convex lower bounding* functions $\hat{h}_{+,j}^{nonc}, \hat{h}_{-,j}^{nonc}, \hat{g}_k^{nonc}$. Because (23.4) is convex its global minimum within some box constraints can be routinely found with any local optimization algorithm (e.g., MINOS 5.4 (Murtagh and Saunders, 1983)) and will always underestimate the global minimum of (23.3) within the same box constraints. Therefore, if the solution of (23.4) inside some rectangular region is strictly positive, then the solution of (23.3) inside the same rectangular domain will also be strictly positive. A strictly positive solution for (23.3) implies that the slack variable s cannot be driven to zero, and thus (23.1) is guaranteed not to have any solutions inside the rectangular region at hand. This provides a mechanism for fathoming (eliminating) parts of the target region which are guaranteed not to contain any solutions. If on the other hand, the global minimum of (23.4) is negative then (23.3) may or may not involve a solution with a zero slack variable and therefore no deduction can be drawn regarding the existence or not of solutions for (23.1) inside the current rectangular domain. In this case, further partitioning of the current rectangular region is required until the global minimum of (23.4) becomes positive or a feasible point for (23.3) is found (convergence).

Based on Property 23.2.3 which demands that the convex lower bounding functions $\hat{h}_{+,j}^{nonc}, \hat{h}_{-,j}^{nonc}, \hat{g}_k^{nonc}$ must be tight, the maximum separation between the original functions and the convex underestimators can become arbitrarily ϵ–small by appropriately reducing the size of the rectangular domain. This implies

that as the current box constraints $[\boldsymbol{x}^L, \boldsymbol{x}^U]$ collapse into a point, the maximum difference ϵ between the original constraint set and its convex relaxation goes to zero. Therefore, any feasible point of problem (23.4) becomes at least ϵ–feasible for problem (23.3) by sufficiently tightening the bounds around this point. Tighter box constraints can be realized by partitioning the current rectangular domain into a number of smaller ones. Note that subdivision is required only for the variables which participate in the nonlinear terms appearing in (23.3).

One way of partitioning is to successively divide the current rectangle into two subrectangles by halving on the middle point of the longest side of the initial rectangle (bisection). At each iteration the lower bound of (23.3) is simply the infimum over all the minima of problem (23.4) in every subrectangle composing the initial rectangle. Therefore, a straightforward (bound improving) way of tightening the lower bound is to halve at each iteration, only the subrectangle responsible for the infimum of the minima of (23.4) over all subrectangles, according to the rules discussed earlier. Clearly, if the global minimum of (23.4) in any subrectangle is strictly greater than zero, then we can safely ignore this subrectangle (fathoming step) because the global minimum of (23.3) cannot be situated inside it. This procedure generates a *nondecreasing* sequence for the lower bound of (23.3) yielding a set of candidate rectangles for containing a solution of (23.1). Convergence is reached when none of the rectangles involve a negative lower bound (no solutions), or when all of the remaining rectangles with negative lower bounds are within the prespecified size tolerance ϵ_r. The basic steps of the proposed algorithm are summarized next.

23.4.2 Algorithmic Steps

STEP 0 - *Initialization*

A size tolerance ϵ_r and a feasibility tolerance ϵ_f are selected and the iteration counter *Iter* is set to one. Appropriate global bounds $\boldsymbol{x}^{LBD}, \boldsymbol{x}^{UBD}$ on $\boldsymbol{x}$ are chosen and local bounds $\boldsymbol{x}^{L,Iter}, \boldsymbol{x}^{U,Iter}$ for the first iteration are set to be equal to the global ones. Finally, an initial point $\boldsymbol{x}^{c,Iter}$ that satisfies the linear equalities and convex inequalities of (23.3) is selected.

STEP 1 - *Feasibility and Convergence Check*

If the maximum violation of all nonconvex constraints of (23.3) calculated at the current point $\boldsymbol{x}^{c,Iter}$ for $(s = 0)$ is less than ϵ_f,

$$\max\left\{\max_{j\in\mathcal{N}_{noncE}} \left|h_j^{nonc}(\boldsymbol{x}^{c,Iter})\right|, \max_{k\in\mathcal{N}_{noncI}} g_k^{nonc}(\boldsymbol{x}^{c,Iter})\right\} \leq \epsilon_f$$

then the point $\boldsymbol{x}^{c,Iter}$ is an ϵ_f–solution of the original problem. Fathom current rectangle if its diagonal is less than ϵ_r,

$$\|\boldsymbol{x}^{U,Iter} - \boldsymbol{x}^{L,Iter}\| \leq \epsilon_r.$$

and proceed to STEP 4. Otherwise, continue with STEP 2.

STEP 2 - *Partitioning of Current Rectangle*

The current rectangle $\left[\boldsymbol{x}^{L,Iter}, \boldsymbol{x}^{U,Iter}\right]$ is partitioned into the following two rectangles ($r = 1, 2$):

$$\begin{bmatrix} x_1^{L,Iter} & x_1^{U,Iter} \\ \vdots & \vdots \\ x_{l^{Iter}}^{L,Iter} & \dfrac{\left(x_{l^{Iter}}^{L,Iter} + x_{l^{Iter}}^{U,Iter}\right)}{2} \\ \vdots & \vdots \\ x_N^{L,Iter} & x_N^{U,Iter} \end{bmatrix}, \quad \begin{bmatrix} x_1^{L,Iter} & x_1^{U,Iter} \\ \vdots & \vdots \\ \dfrac{\left(x_{l^{Iter}}^{L,Iter} + x_{l^{Iter}}^{U,Iter}\right)}{2} & x_{l^{Iter}}^{U,Iter} \\ \vdots & \vdots \\ x_N^{L,Iter} & x_N^{U,Iter} \end{bmatrix}$$

where l^{Iter} corresponds to the variable with the longest side in the initial rectangle,

$$l^{Iter} = \arg\max_i \left(x_i^{U,Iter} - x_i^{L,Iter}\right)$$

STEP 3 - *Solution of Convex Problems Inside Subrectangles*

Solve the convex optimization problem (23.4) in both subrectangles ($r = 1, 2$) by using any convex nonlinear solver (e.g., MINOS 5.4 (Murtagh and Saunders, 1983)). If the solution $s_{sol}^{r,Iter}$ is negative, then it is stored along with the value of the variables $\boldsymbol{x}$ at the solution point $\boldsymbol{x}_{sol}^{r,Iter}$. If $s_{sol}^{r,Iter}$ is strictly positive, then the element $(r, Iter)$ is fathomed.

STEP 4 - *Update Iteration Counter Iter and Lower Bound s^{LBD}*

The iteration counter is increased by one,

$$Iter \longleftarrow Iter + 1$$

and the lower bound s^{LBD} is updated to be the minimum solution over the stored ones from previous iterations. Furthermore, the selected solution is erased from the stored set.

$$\begin{aligned} s^{LBD} &= s_{sol}^{r',Iter'} \\ \text{where} \quad s_{sol}^{r',Iter'} &= \min_{r,I} s_{sol}^{r,I}, \quad r = 1, 2, \quad I = 1, \ldots, Iter - 1. \end{aligned}$$

STEP 5 - *Update Current Point $\boldsymbol{x}^{c,Iter}$ and Current Bounds $\boldsymbol{x}^{L,Iter}, \boldsymbol{x}^{U,Iter}$*

The current point is selected to be the solution point of the previously found minimum solution in STEP 4,

$$\boldsymbol{x}^{c,Iter} = \boldsymbol{x}_{sol}^{r',Iter'}$$

and the current rectangle becomes the subrectangle containing the previously found solution,

$$\left[\boldsymbol{x}^{L,Iter}, \boldsymbol{x}^{U,Iter}\right] = \begin{bmatrix} x_1^{L,Iter'} & x_1^{U,Iter'} \\ \vdots & \vdots \\ x_{l^{Iter'}}^{L,Iter'} & \dfrac{\left(x_{l^{Iter'}}^{L,Iter'} + x_{l^{Iter'}}^{U,Iter'}\right)}{2} \\ \vdots & \vdots \\ x_N^{L,Iter'} & x_N^{U,Iter'} \end{bmatrix}, \quad \text{if} \quad r' = 1$$

$$\left[\boldsymbol{x}^{L,Iter}, \boldsymbol{x}^{U,Iter}\right] = \begin{bmatrix} x_1^{L,Iter'} & x_1^{U,Iter'} \\ \vdots & \vdots \\ \dfrac{\left(x_{l^{Iter'}}^{L,Iter'} + x_{l^{Iter'}}^{U,Iter'}\right)}{2} & x_{l^{Iter'}}^{U,Iter'} \\ \vdots & \vdots \\ x_N^{L,Iter'} & x_N^{U,Iter'} \end{bmatrix}, \quad \text{if} \quad r' = 2$$

STEP 7 - *Check for Convergence*

$$\text{If} \quad s_{LBD} \leq 0, \text{ then return to STEP 1.}$$

Otherwise, terminate.

23.4.3 Outline of Convergence Proof

Mathematical proof that the proposed procedure is guaranteed to converge to a set of disjoint rectangles containing all global minimum solutions of (23.3) is given based on the analysis of a standard deterministic global optimization algorithm presented in Horst and Tuy (1990). Because the employed branch and bound technique fathoms only rectangles *guaranteed not to contain any global minima of (23.3)* , no solutions of (23.3) which are at least ϵ_r apart are missed. A sufficient condition for the proposed branch and bound algorithm to be convergent to the global minima, requires that the bounding operation must be *consistent* and the selection operation *bound improving.*

A bounding operation is called *consistent* if (i) at every step any unfathomed partition can be further refined, and (ii) for any infinitely decreasing sequence of successively refined partition elements the gap between the lower and upper bounds goes to zero as the iterations go to infinity. Due to properties 23.2.1, 23.2.2, and 23.2.3, the gap between the lower and upper bound for any partition element goes to zero as the size of the partition element goes to zero. Furthermore, the employed bisection subdivision process (bisection along the longest side) is exhaustive because the size of an infinitely partitioned element goes to zero. Therefore, the bounding operation is consistent. Also, the employed

selection operation is *bound improving* because the partition element where the actual lower bound is attained is selected for further partition in the immediately following iteration. Therefore according to Theorem IV.3. in Horst and Tuy (1990) the employed global optimization algorithm is *convergent* to the global minima of (23.3).

23.5 COMPUTATIONAL STUDIES

In this section, several example problems are addressed to illustrate the applicability and efficiency of the deterministic global optimization approach for the enclosure of all solutions of nonlinear constrained systems of equations. These problems include standard test examples, as well as steady–state identification in reaction networks, multiple steady states in flowsheets and CSTRs, and circuit design problems. The presented branch and bound convex lower bounding algorithm has been implemented in GAMS (Brooke *et al.*, 1992) and computational times are reported for all examples on a HP–730 workstation with size and feasibility tolerances of 10^{-4} (Maranas and Floudas (1995)). In the GAMS implementation, small fixed values of the parameters α are used. It has also been implemented in C as part of the αBB global optimization code that was discussed in Chapter 13. We present results of the αBB implementation for a selective number of example problems described in the doctoral thesis of Adjiman (1998). In the αBB implementation, the scaled Gerschgorin theorem with $\boldsymbol{d} = \boldsymbol{x}^U - \boldsymbol{x}^L$ is employed for the rigorous calculation of the α parameters.

Example 1. This example involves locating the stationary points of the Himmelblau function as described in Reklaitis and Ragsdell (1983). It is a standard test problem which is used to check the ability of the method to generate all nine solutions of the problem irrespective of the selection of the initial point.

$$\begin{aligned} 4x_1^3 + 4x_1x_2 + 2x_2^2 - 42x_1 - 14 &= 0 \\ 4x_2^3 + 2x_1^2 + 4x_1x_2 - 26x_2 - 22 &= 0 \end{aligned}$$

$$\begin{aligned} -5.0 \leq x_1 \leq 5.0 \\ -5.0 \leq x_2 \leq 5.0. \end{aligned}$$

First, the change of variables

$$y_1 = 1.0 + \frac{9}{10}(x_1 + 5.0), \quad y_2 = 1.0 + \frac{9}{10}(x_2 + 5.0)$$

is performed which ensures that all variables will be positive. This results in the following system of equations:

$$\begin{aligned} \frac{2}{77}y_1^3 - \frac{30}{77}y_1^2 - \frac{20}{77}y_2 + \frac{2}{77}y_1y_2 + \frac{1}{77}y_2^2 + \frac{17}{11}y_1 - 1 &= 0 \\ \frac{2}{121}y_2^3 - \frac{30}{121}y_2^2 + \frac{2}{121}y_1y_2 + \frac{127}{121}y_2 + \frac{1}{121}y_1^2 - \frac{20}{121}y_1 - 1 &= 0 \end{aligned}$$

$$\begin{array}{rcccl} 1.0 & \leq & y_1 & \leq & 10.0 \\ 1.0 & \leq & y_2 & \leq & 10.0. \end{array}$$

Then, the exponential variable transformation is employed for deriving a convex relaxation of the original problem. The problem is solved with the GAMS implementation after 197 iterations and 10.89 seconds of CPU time and provides all nine solutions,

$$\begin{array}{rclrcl} x_1^* & = & -0.2709, & x_2^* & = & -0.9230 \\ x_1^* & = & -0.1279, & x_2^* & = & -1.9538 \\ x_1^* & = & 3.5844, & x_2^* & = & -1.8481 \\ x_1^* & = & 3.3852, & x_2^* & = & 0.0739 \\ x_1^* & = & 3.0000, & x_2^* & = & 2.0000 \\ x_1^* & = & 0.0867, & x_2^* & = & 2.8843 \\ x_1^* & = & -2.8051, & x_2^* & = & 3.1313 \\ x_1^* & = & -3.0730, & x_2^* & = & -0.0814 \\ x_1^* & = & -3.7793, & x_2^* & = & -3.2832 \end{array}$$

are located from any initial point. The αBB implementation required 129 iterations and 16 CPU seconds.

Example 2. This example addresses the identification of the steady states (single or multiple) of a process flow diagram for the production of dibutyl phthalate (Seader *et al.*, 1990). The formulation of this problem involves a number of bilinear terms.

$$\begin{aligned} 5.5(x_4 - x_2)(x_2 + x_3) - x_2(x_1 - x_2 - x_3) &= 0 \\ 3.5(0.888x_1 - x_3 - x_4)(x_2 + x_3) - x_3(x_1 - x_2 - x_3) &= 0 \\ 5.5(0.0986x_1 - x_2 - x_5 - 0.01(1.098x_1 - x_2 - 9x_4 - x_5 - x_6 + x_7)) & \\ (x_5 + x_6) - x_5(1.098x_1 - x_2 - 9x_4 - x_5 - x_6 + x_7) &= 0 \\ 3.5(0.986x_1 - 9x_4 - x_5 - x_6 - 0.0986x_1 & \\ +0.01(1.098x_1 - x_2 - 9x_4 - x_6 + x_7)) & \\ (x_5 + x_6) - x_6(1.098x_1 - x_2 - 9x_4 - x_5 - x_6 + x_7) &= 0 \\ x_4(x_1 - x_2 - x_3)^2 - 2056.4(0.098x_1 - x_4)^2 &= 0 \\ 0.61177 - (0.0986x_1 - x_4) + 0.01(1.098x_1 - x_2 - 9x_4 - x_5 - x_6 + x_7) &= 0 \\ \frac{0.04}{98.01}(74.12(0.986x_1 - 10x_4) + 222.24(0.0986x_1 - x_4) & \\ +18(x_4 - x_2) + 278.84x_4 + 98.09(0.0135x_1)) - 0.0136x_1 &= 0 \end{aligned}$$

$$0.0001 \leq y_i \leq 100.0, \quad i = 1, \ldots, 5.$$

Using the exponential variable transformation and the GAMS implementation it was shown that there exists no solution to this problem after 120 iterations and 50 seconds of CPU time. Note that the solution reported in Seader *et al.* (1990) did not satisfy our feasibility tolerance of 10^{-4}.

Example 3. This example by Bullard and Biegler (1991) addresses a badly scaled system of equations which have troubled existing solvers. It involves exponential as well as bilinear terms.

$$\begin{aligned} 10^4 x_1 x_2 - 1 &= 0 \\ \exp(-x_1) + \exp(-x_2) - 1.001 &= 0 \end{aligned}$$

$$\begin{array}{ccccc} 0 & \leq & x_1 & \leq & 100 \\ 0 & \leq & x_2 & \leq & 100. \end{array}$$

The bilinear terms $x_1 x_2, -x_1 x_2$ are underestimated based on the analysis in section 23.3. The terms $\exp(-x_1), \exp(-x_2)$ are convex. The terms $-\exp(-x_1)$, and $-\exp(-x_2)$ are univariate concave terms and are convex lower bounded with a line segment.

In 32 iterations and 1.5 seconds of CPU time, using the GAMS implementation, it was shown that

$$x_1^* = 0.0000145067, \quad x_2^* = 6.89335287$$

is a unique solution to the problem. Note that, the second solution $x_1^* = 0.00001098$, $x_2^* = 9.106$ reported in Bullard and Biegler (1991) does not satisfy the nonlinear equations. The αBB implementation required 54 iterations and 1.5 CPU seconds.

Example 4. This test example proposed by Ferraris and Tronconi (1986), involves a combination of trigonometric and exponential terms. Again this example is included because it has been reported to be extremely difficult to solve.

$$\begin{aligned} 0.5\sin(x_1 x_2) - 0.25 x_2/\pi - 0.5 x_1 &= 0 \\ (1 - 0.25/\pi)(\exp(2x_1) - e) + e x_2/\pi - 2e x_1 &= 0 \end{aligned}$$

$$\begin{array}{ccccc} 0.25 & \leq & x_1 & \leq & 1 \\ 1.5 & \leq & x_2 & \leq & 6.28. \end{array}$$

The α–based underestimation was chosen to address the convex lower bounding of the term $\sin(x_1 x_2)$. The eigenvalues of this term are equal to:

$$\lambda_{1,2} = \frac{-\left(x_1^2 + x_2^2\right)\sin(x_1 x_2)}{2} \pm \frac{\sqrt{4 - 8x_1 x_2 \sin(x_1 x_2)\cos(x_1 x_2) + \left[\left(x_1^2 + x_2^2\right)^2 - 4\right]\sin(x_1 x_2)^2}}{2}.$$

A lower bound on this expression is then:

$$\lambda_{min} \geq -\max\left((x_1^L)^2, (x_1^U)^2\right) - \max\left((x_2^L)^2, (x_2^U)^2\right).$$

Therefore,

$$\alpha = \frac{\max\left((x_1^L)^2, (x_1^U)^2\right) + \max\left((x_2^L)^2, (x_2^U)^2\right)}{2}.$$

Using the GAMS implementation, the following two solutions were found to this problem:

$$\begin{array}{rclrcl} x_1^* & = & 0.29944869, & x_2^* & = & 2.83692777 \\ x_1^* & = & 0.50000000, & x_2^* & = & 3.14159265 \end{array}$$

in 45 iterations and 2.0 seconds of CPU time. The αBB implementation required 59 iterations and 1.6 CPU seconds. Note that, both solutions were missed in Bullard and Biegler (1991).

Example 5. This test problem is Brown's almost linear system (Kearfott and Novoa, 1990). It is a difficult problem to solve because it involves two distinct solutions which are very close to each other.

$$\begin{aligned} 2x_1 + x_2 + x_3 + x_4 + x_5 - 6 &= 0 \\ x_1 + 2x_2 + x_3 + x_4 + x_5 - 6 &= 0 \\ x_1 + x_2 + 2x_3 + x_4 + x_5 - 6 &= 0 \\ x_1 + x_2 + x_3 + 2x_4 + x_5 - 6 &= 0 \\ x_1 x_2 x_3 x_4 x_5 - 1 &= 0 \end{aligned}$$

$$-2 \leq x_i \leq 2, \quad i = 1, \ldots, 5.$$

This system exhibits two solutions:

$$\begin{array}{rclrcl} x_1^* & = & 1.000, & x_1^* & = & 0.916 \\ x_2^* & = & 1.000, & x_2^* & = & 0.916 \\ x_3^* & = & 1.000, & x_3^* & = & 0.916 \\ x_4^* & = & 1.000, & x_4^* & = & 0.916 \\ x_5^* & = & 1.000, & x_5^* & = & 1.418. \end{array}$$

The α parameter was used to construct convex lower bounds for the nonconvex constraint. Using the GAMS implementation, the following convergence rates are obtained for different values of α.

	Iter	*CPU*	
$\alpha = 1000$	352	22.16	(*Convergence*)
$\alpha = 100$	112	7.42	(*Convergence*)
$\alpha = 10$	37	2.26	(*Convergence*)
$\alpha = 5$	12	0.69	(*Convergence*)
$\alpha = 1$	7	0.35	(*Convergence*)
$\alpha = 0.1$	–	–	(*Failure*)

Note that the total number of iterations remains relatively small even for very large values of α. Moreover, a value of α of only one appears to be sufficient. Using the αBB implementation that employs a rigorous calculation of the α parameters via the scaled Gerschgorin theorem, requires 57 iterations and 15.5 CPU seconds.

Example 6. This example, presented in detail by Kubicek *et al.* (1980), is a model of two continuous, non–adiabatic, stirred tank reactors in series at steady state with recycle and an exothermic, first order, irreversible reaction. The modelling equations of this system involve products between linear and exponential terms and are:

$$\begin{aligned}
(1-R)(1-y_1)\exp\left(\frac{10\phi_1}{1+10\phi_1/\gamma}\right) - y_1 &= 0\\
(1-R)D(1-y_1)\exp\left(\frac{10\phi_1}{1+10\phi_1/\gamma}\right) - 10(1+\beta_1)\phi_1 + \beta_1\phi_{c1} &= 0\\
y_1 - y_2 + (1-R)(1-y_2)\exp\left(\frac{10\phi_2}{1+10\phi_2/\gamma}\right) &= 0\\
10\phi_1 - 10(1+\beta_2)\phi_2 + (1-R)D(1-y_2)\exp\left(\frac{10\phi_2}{1+10\phi_2/\gamma}\right) + \beta_2\phi_{c2} &= 0.
\end{aligned}$$

The variables are:

$$\begin{aligned}
y_i &= \text{Reactant conversion in reactor } i\text{=1,2}\\
\phi_i &= \text{Dimensionless temperature in reactor } i\text{=1,2.}
\end{aligned}$$

The parameters are:

$$\begin{aligned}
\gamma &= 1000, \text{ Dimensionless activation energy}\\
D &= 22, \text{ Dimensionless adiabatic temperature rise}\\
\phi_{c1} = \phi_{c2} &= 0, \text{ Dimensionless coolant temperatures}\\
\beta_1 = \beta_2 &= 2, \text{ Dimensionless heat transfer coefficient in reactors 1,2}\\
0 \le R &\le 1, \text{ Recycle ratio.}
\end{aligned}$$

Elimination of the y_i variables yields the following compact system of equations in two variables ϕ_1, ϕ_2.

$$\begin{aligned}
(1-R)\left[\frac{D}{10(1+\beta_1)} - \phi_1\right]\exp\left(\frac{10\phi_1}{1+10\phi_1/\gamma}\right) - \phi_1 &= 0\\
\phi_1 - (1+\beta_2)\,\phi_2 + (1-R)\left[D/10 - \beta_1\phi_1 - (1+\beta_2)\phi_2\right]\exp\left(\frac{10\phi_2}{1+10\phi_2/\gamma}\right) &= 0
\end{aligned}$$

$$0 \le \phi_i \le 1, \quad i = 1,2.$$

Note that the nonlinear terms,

$$\exp\left(\frac{10\phi_i}{1+10\phi_i/\gamma}\right), \quad \phi_i\exp\left(\frac{10\phi_i}{1+10\phi_i/\gamma}\right), \quad i = 1,2$$

R	Iter-GAMS	CPU-GAMS	Iter-αBB	CPU-αBB	# Sol's
0.935	65	3.8	52	2.7	1
0.940	82	4.7	51	2.7	1
0.945	437	18.8	117	5.5	3
0.950	340	15.7	161	6.9	5
0.955	317	14.8	162	6.6	5
0.960	539	25.1	216	8.6	7
0.965	267	12.6	156	6.3	5
0.970	270	13.0	154	6.1	5
0.975	214	10.6	155	6.3	5
0.980	240	11.1	149	6.1	5
0.985	228	11.6	141	11.6	5
0.990	65	3.5	60	3.6	1
0.995	23	1.3	41	1.3	1

Table 23.1. Number of solutions, iterations, and CPU requirements for different recycle ratios

appearing in this system are convex and thus are left unchanged in the lower bounding problem. The negative of these terms are concave functions and are convex lower bounded with a line segment. The terms,

$$\pm\phi_1 \exp\left(\frac{10\phi_2}{1+10\phi_2/\gamma}\right),$$

are products of two univariate functions and are underestimated based on the analysis in subsection 23.3.

The problem was solved for different values of the recycle ratio R, starting with R=0.935 and ending at R=0.995 with 0.005 increments. For each value of R, the total number of distinct solutions, the total number of iterations and CPU requirements are shown in Table 23.1 for both the GAMS and the αBB implementation. The detailed solutions for each value of R can be found in the doctoral thesis of Adjiman (1998).

Example 7. This example involves the solution of a circuit design problem (Ratschek and Rokne, 1993) described by the following set of equations.

$$\begin{aligned}(1 - x_1x_2)\, x_3 \left\{\exp\left[x_5\left(g_{1k} - g_{3k}x_7 10^{-3} - g_{5k}x_8 10^{-3}\right)\right] - 1\right\} & \\ -g_{5k} + g_{4k}x_2 &= 0, \\ k = 1, \ldots, 4 & \end{aligned}$$

$$\begin{aligned}(1 - x_1x_2)\, x_4 \left\{\exp\left[x_6\left(g_{1k} - g_{2k} - g_{3k}x_7 10^{-3} + g_{4k}x_9 10^{-3}\right)\right] - 1\right\} & \\ -g_{5k}x_1 + g_{4k} &= 0, \\ k = 1, \ldots, 4 & \end{aligned}$$

$$x_1x_3 - x_2x_4 = 0$$

$$0 \leq x_i \leq 10, \quad i = 1, \ldots, 9$$

where

	$k=1$	$k=2$	$k=3$	$k=4$
g_{1k}	0.4850	0.7520	0.8690	0.9820
g_{2k}	0.3690	1.2540	0.7030	1.4550
g_{3k}	5.2095	10.0677	22.9274	20.2153
g_{4k}	23.3037	101.7790	111.4610	191.2670
g_{5k}	28.5132	111.8467	134.3884	211.4823

This problem is addressed because it has been reported to involve extraordinary sensitivities to small perturbations in the variables leading to numerical stability problems for gradient–based solvers. The single positive solution of the problem,

$$\begin{array}{lllllll} x_1^* & = & 0.899999, & x_6^* & = & 7.999693 \\ x_2^* & = & 0.449987, & x_7^* & = & 5.000031 \\ x_3^* & = & 1.000006, & x_8^* & = & 0.999988 \\ x_4^* & = & 2.000069, & x_9^* & = & 2.000052 \\ x_5^* & = & 7.999971, & & & \end{array}$$

was first reported by Ratschek and Rokne (1993). We utilize the α parameter to determine a convex lower bound for the various nonlinear terms. Computational requirements for various values of α using the GAMS implementation are as follows:

	Iter	*CPU(sec)*	
$\alpha = 0.1$	1645	987.91	(*Convergence*)
$\alpha = 0.01$	212	143.41	(*Convergence*)
$\alpha = 0.001$	–	–	(*Failure*).

Note that these CPU requirements are only a fraction of the ones reported in Ratschek and Rokne (1993).

Furthermore, after relaxing the variable bounds to

$$-10 \leq x_i \leq 10, \quad i = 1, \ldots, 9$$

a second solution was found

$$\begin{array}{rcrcr} x_1^* & = & 0.823226, & x_6^* & = & -2.765092 \\ x_2^* & = & -0.553286, & x_7^* & = & 6.046646 \\ x_3^* & = & 0.671878, & x_8^* & = & 0.975940 \\ x_4^* & = & -0.999677, & x_9^* & = & -1.708489 \\ x_5^* & = & 8.854525. & & & \end{array}$$

which was missed in all previous attempts, reported in the literature, at solving this problem.

Example 8. This example is concerned with finding all multiple steady states of a nonisothermal CSTR reactor (Smith, 1985) and the effect of small changes in the reaction enthalpy ΔH on the number of steady states. The single independent variable of the problem is the reactor temperature T.

$$\frac{\rho C_p (T - T_o)}{C_o \Delta H} + \theta A \exp(-E/RT)[1 + \theta A \exp(-E/RT)] = 0$$

$$0 \leq T \leq 10$$

where

$$\begin{array}{rcl} \rho & = & 1.0\ (gmol/cm^3), \text{ fluid density} \\ C_p & = & 1.0\ (cal/gmol), \text{ specific heat} \\ C_o & = & 3.0\ 10^{-3}\ (gmol/cc), \text{ inlet concentration} \\ T_o & = & 298\ (K), \text{ inlet temperature} \\ \theta & = & 3000\ (sec), \text{ average residence time} \\ A & = & 4.48\ 10^6\ (l/gmol\ sec), \text{ frequency factor} \\ E & = & -15,000\ (cal/gmol), \text{ activation energy} \\ R & = & 1.98725\ (cal/gmol\ K), \text{ gas constant} \end{array}$$

After defining the constants,

$$a = \frac{\rho C_p T_o}{C_o \Delta H} < 0, \quad b = \theta A > 0, \quad c = -E/B > 0$$

and multiplying the equation with $[1 + \theta A \exp(-E/RT)] > 0$ we have:

$$\left(\frac{b}{T_o}\right) T \exp(c/T) - \frac{b(1 + aT_o)}{aT_o} \exp(c/T) + \left(\frac{T}{T_o}\right) - 1 = 0.$$

Note that, the terms

$$f(T) = T \exp(c/T), \quad g(T) = \exp(c/T)$$

are convex since

$$f''(T) = \frac{c^2 \exp(c/T)}{T^3} > 0, \quad g''(T) = \frac{c(c+2T)\exp(c/T)}{T^4} > 0.$$

Therefore, $-f(T), -g(T)$ are concave and are underestimated with a line segment. Three different values for the reaction enthalpy ΔH in $cal/gmol$ are examined and the results of the GAMS implementation are as follows:

	$Iter$	$CPU(sec)$	T_1^*	T_2^*	T_3^*
$\Delta H = -50,000.0$	46	2.20	300.4239	347.4099	445.4874
$\Delta H = -35,958.0$	54	2.55	299.6330	380.4600	380.5068
$\Delta H = -35,510.3$	56	2.66	299.6096	–	–

The αBB implementation for the three aforementioned cases required 79, 94 and 63 iterations with CPU times of 2.5, 3.4 and 2.5 seconds, respectively.

Summary

In this chapter we discussed a deterministic branch and bound type algorithm for enclosing all ϵ–global solutions of constrained systems of nonlinear equations (Maranas and Floudas, 1995). The approach is based on the one–to–one correspondence between the multiple solutions of the nonlinear systems and the multiple global minima with a zero objective value for the resulting nonconvex optimization problem. All multiple ϵ–global minima of the nonconvex optimization problem are enclosed based on the construction of upper bounds based on function evaluations and lower bounds on the global minimum solution through the solution of convex relaxations of the constraint set. Based on the form of the participating functions, a number of alternative techniques for constructing this convex relaxation are proposed. In particular, by taking advantage of the properties of products of univariate functions, customized convex lower bounding functions are introduced for a large number of expressions that are, or can be, transformed into products of univariate functions. The presented branch and bound approach is guaranteed to enclose all ϵ–solutions of (23.1) within arbitrarily small rectangles in a finite number of iterations. A number of example problems from many areas of research have been addressed and in all cases, convergence to all multiple solutions was achieved with reasonable computational effort. Furthermore, in certain cases new solutions were identified. Additional computational studies can be found in Maranas and Floudas (1995) and in the doctoral thesis of Adjiman (1998). Explicit mathematical formulations of the examples considered here and additional examples can be found in the recent book of test problems for local and global optimization by Floudas *et al.* (1999).

The proposed deterministic global optimization framework for the enclosure of all solutions of nonlinear constrained systems of equations has been applied to several important application areas. These include (i) the enclosure of all homogeneous azeotropes (Harding *et al.*, 1997) which is discussed in the following chapter, (ii) the enclosure of all heterogeneous and reactive

azeotropes (Harding and Floudas, 1999a), and (iii) the enclosure of all transition states of potential energy surfaces of triatomic molecules and oligopeptides and the study of reaction pathways (Westerberg and Floudas, 1999a; Westerberg and Floudas, 1999b).

24 LOCATING ALL HOMOGENEOUS AZEOTROPES

In this chapter, we study the enclosure of all homogeneous azeotropes in multicomponent mixtures. Section 24.1 introduces the reader to the importance of locating all azeotropes problem and outlines the previous work. Section 24.2 defines the problem. Section 24.3 introduces the general mathematical formulation. Section 24.4 focuses on the activity coefficient models of the Wilson equation, the NRTL equation, the UNIQUAC equation, and the UNIFAC equation, presents the resulting mathematical models and develops the underestimating functions for these models. Finally, section 24.5 presents selective computational studies for all aforementioned activity coefficient models. The material presented in this chapter is based on the work of Harding *et al.* (1997).

24.1 INTRODUCTION

The ability to predict whether a given mixture will form one or more azeotropes and to calculate the conditions and compositions of each azeotrope is essential if one wants to model separation processes. Furthermore, it is necessary to calculate the effects of temperature and pressure on the composition of an azeotrope for process design applications.

Many thermodynamic models have been proposed to predict the phase behavior of nonideal mixtures. Unfortunately, the accuracy of these models is not uniform over a wide range of mixtures. One useful way of testing the accuracy of a model for a given mixture is to compare the compositions of the azeotropes

C. A. Floudas, *Deterministic Global Optimization*
© Springer Science+Business Media Dordrecht 2000

predicted by the model with those determined by experiment. A method for calculating the azeotropes of a mixture must be robust, and must be able to guarantee that all possible azeotropes can be found.

Despite the considerable interest in the area of predicting phase equilibria for chemical mixtures, relatively few methods for the prediction of azeotropes have been reported. This is because the task of finding the pressure, temperature, and composition of *all* azeotropes is an especially difficult one, due to the highly nonlinear form of the equations which constitute the thermodynamic models.

Most of the previous work reported in the literature has been limited to calculating homogeneous azeotropes. Aristovich and Stepanova (1970) calculated ternary homogeneous azeotropes using the Wilson model under isothermal conditions. Teja and Rowlinson (1973) calculated homogeneous azeotropes of binary mixtures using an equation of state as the thermodynamic model. Their approach was to fix temperature and vary composition and volume until the thermodynamic equilibrium conditions were satisfied. Wang and Whiting (1986) also used an equation of state to calculate homogeneous azeotropes for binary mixtures. Fidkowski *et al.* (1993) have presented an interesting homotopy continuation method for finding homogeneous azeotropes.

Chapman and Goodwin (1993) presented a search method for finding homogeneous and heterogeneous azeotropes which uses a Levenberg-Marquardt algorithm to find homogeneous azeotropes and then checks the stability of each solution with the tangent plane criterion described by Michelsen (1981). A solution which is found to be unstable is then used as the starting point for a new search for an heterogeneous azeotrope. This method is limited by the fact that heterogeneous azeotropes do not necessarily correspond to an unstable homogeneous azeotrope solution and vice versa. It cannot find azeotropes for which one or more components have a vapor phase mole fraction of zero. An excellent review on nonideal distillation, including a discussion on the computation of azeotropes has recently been published by Widagdo and Seider (1996).

The primary objective in this chapter is to propose a new method for determining all homogeneous azeotropes of a non-reacting mixture for several different thermodynamic models. The vapor phase is modeled as an ideal gas at low pressure, and activity coefficient models are used for the liquid phase. The activity coefficient models examined are: the Wilson equation, the NRTL equation, the UNIQUAC equation, and the UNIFAC equation. We follow the approach introduced in Chapter 23 and formulate the problem as a deterministic global optimization problem in which each global minimum solution corresponds to an homogeneous azeotrope.

24.2 PROBLEM DESCRIPTION

In this section, the fundamentals of azeotropy will be summarized and the thermodynamic equations which determine the existence of azeotropes will be presented. Finally, the problem of enclosing all homogeneous azeotropes will be formulated as a global optimization problem in which the objective is to enclose all global solutions.

24.2.1 Azeotropy

In order to develop a method for finding all azeotropes of a mixture, it is essential to first determine the thermodynamic conditions for azeotropy. Azeotropes occur in a boiling mixture of one or more liquid phases when the composition of the vapor phase is the same as the overall composition of the liquid phase(s).

Azeotropes can be classified by two categories: order and type. The order of the azeotrope describes the number of species that participate in the azeotrope. For example, a mixture of compounds A, B, and C may have a ternary azeotrope ABC in which all three components participate, and three binary azeotropes, AB, AC, and BC. The type of azeotrope reflects the number of phases present at the point of azeotropy. When a vapor phase and only one liquid phase are present, the azeotrope is termed an *homogeneous azeotrope.* In the case of homogeneous azeotropy, application of the condition that the overall composition of the phases are identical produces the familiar equation:

$$x_i = y_i \qquad \forall\, i \in N \tag{24.1}$$

where x_i is the mole fraction of component i in the liquid phase, y_i is the mole fraction of component i in the vapor phase, and N is the set of components. The second type of azeotrope is an *heterogeneous azeotrope.* Heterogeneous azeotropes occur when a vapor phase and two or more liquid phases are present at the point of azeotropy. In this case, the composition of the vapor phase must be equal to the *overall* composition of the liquid phases. In general, a mixture of N components and M phases may have both homogeneous and heterogeneous azeotropes. The third type of azeotrope is a *reactive azeotrope* which occurs when we have azeotropes in the presence of reaction(s).

24.2.2 Thermodynamic Conditions for Azeotropy

There are three thermodynamic conditions that a system must meet in order for an azeotrope to exist. These conditions are: 1) equilibrium, 2) the composition of vapor phase must be identical to the overall composition of the liquid phase(s), and 3) the mole fractions of the components in each phase must sum to unity and must be non-negative.

The equilibrium condition requires that the chemical potential of each component must be the same in all phases. Since an homogeneous azeotropic system contains a vapor phase (V), and only one liquid phase (L), this condition can be written:

$$\mu_i^V = \mu_i^L \qquad \forall\, i \in N \tag{24.2}$$

where μ_i^V and μ_i^L represent the chemical potential of component i in the vapor and liquid phases. From the definition of the fugacity of component i in a mixture, $\hat{f}_i$,

$$\hat{f}_i^V = \hat{f}_i^L \qquad \forall\, i \in N \tag{24.3}$$

hence,

$$y_i \hat{\phi}_i^V P = x_i \gamma_i^L f_i^L. \tag{24.4}$$

The symbol $\hat{\phi}_i^V$ represents the mixture fugacity coefficient of component i in the vapor phase. For the liquid phase, γ_i^L is the activity coefficient, and f_i^L is the fugacity of pure component i in the liquid phase. Rearranging equation (24.4) gives,

$$\frac{y_i}{x_i} = \frac{\gamma_i^L f_i^L}{\hat{\phi}_i^V P} \qquad \forall\, i \in N. \tag{24.5}$$

At low pressure, the vapor phase can be modeled as an ideal gas, for which $\hat{\phi}_i^V = 1$, and for the liquid phase the fugacity is equal to $f_i^L = \phi_i^{sat} P_i^{sat} (PF)_i$. But, for an ideal gas, $\phi_i^{sat} = 1$, and $(PF)_i = 1$. Therefore, we have :

$$\frac{y_i}{x_i} = \frac{\gamma_i^L P_i^{sat}}{P}. \tag{24.6}$$

Now all of the nonidealities of the system are confined to the activity coefficient term. For binary homogeneous azeotropes, negative deviations from Raoult's law ($\gamma_i^L < 1$) result in a maximum-boiling azeotrope, also called a minimum-pressure azeotrope. Positive deviations result in a minimum-boiling (maximum-pressure) azeotrope. Ternary and higher order azeotropes may also occur at saddle points in the temperature (or pressure) vs. composition surface. An heterogeneous azeotrope only occurs when positive deviations from Raoult's law are very large ($\gamma_i^{L_j} \gg 1$) where j represents each liquid phase $j \in M$, and the vapor-liquid equilibrium surface overlaps the liquid-liquid equilibrium surface.

Note that for an homogeneous azeotrope, the vapor phase and the single liquid phase have the same composition. In the heterogeneous case, the azeotropy condition requires that the composition of the vapor phase is identical to the overall composition of the two or more liquid phases. In the case of an homogeneous azeotrope, this condition simplifies to:

$$y_i = x_i \qquad \forall\, i \in N. \tag{24.7}$$

The third condition requires that the mole fractions in each phase sum to unity and have values between 0 and 1.

$$\begin{aligned} \sum_{i \in N} y_i = \sum_{i \in N} x_i = 1 \\ 0 \le y_i, x_i \le 1 \qquad \forall\, i \in N. \end{aligned} \tag{24.8}$$

24.3 MATHEMATICAL FORMULATION

To locate all azeotropes, one must find all solutions to the system of nonlinear equations (24.6), (24.7), and (24.8). This chapter uses the approach discussed

in Chapter 23, which reformulates the problem of enclosing all solutions of nonlinear systems of constrained equations into a global optimization problem in which the task is to enclose all global solutions. In this approach, each nonlinear equality is replaced by two inequalities and a single slack variable is introduced. For the location of all homogeneous azeotropes, this corresponds to employing equations (24.6), (24.7), and (24.8) and reformulating them as the following global optimization problem:

$$
\begin{array}{lll}
\min\limits_{\boldsymbol{x},\boldsymbol{y},T,s} & s & \\
\text{s.t.} & Py_i - P_i^{sat}\gamma_i x_i - s \leq 0 & \forall\, i \in N \\
 & -Py_i + P_i^{sat}\gamma_i x_i - s \leq 0 & \forall\, i \in N \\
 & y_i - x_i = 0 & \forall\, i \in N \\
 & \sum\limits_{i\in N} y_i = 1 & \\
 & \sum\limits_{i\in N} x_i = 1 & \\
 & s \geq 0 & \\
 & 0 \leq y_i \leq 1 & \forall\, i \in N \\
 & 0 \leq x_i \leq 1 & \forall\, i \in N.
\end{array}
\tag{24.9}
$$

Remark 24.3.1 Problem (24.9) may have multiple global minima. Each global minimum of Problem (24.9) (where $s^* = 0$) corresponds to an homogeneous azeotrope since when $s = 0$ the constraints (24.6), (24.7), and (24.8) are satisfied. Note that the first two sets of constraints of (24.9) correspond to the nonlinear equations (24.6) of the equilibrium constraint written as two inequalities. In addition, note that the nonlinear term $P_i^{sat}\gamma_i x_i$ appears as both a positive and a negative term. Thus, this term must be nonconvex in at least one of the two constraints. This means that if a local optimization approach is used to solve (24.9), some or all of the global solutions may be missed.

Since $x_i = y_i$ for every i, the system of equations can be simplified by replacing each y_i by x_i. Therefore, the equilibrium condition becomes,

$$x_i(P - P_i^{sat}\gamma_i) = 0 \qquad \forall\, i \in N. \tag{24.10}$$

In order to satisfy this condition, either $x_i = 0$, or $(P - P_i^{sat}\gamma_i) = 0$ for each $i \in N$. If we wish to search only for azeotropes in which *all* components participate (an N-ary azeotrope), the condition can be further simplified to $(P - P_i^{sat}\gamma_i) = 0$ for each $i \in N$.

Because P_i^{sat} and γ_i are usually given as $\ln P_i^{sat}$ and $\ln \gamma_i$, it is convenient to take the natural log of the simplified equilibrium constraint, resulting in:

$$\ln P - \ln P_i^{sat} - \ln \gamma_i = 0 \qquad \forall\, i \in N. \tag{24.11}$$

Then the formulation becomes:

$$\begin{array}{ll} \min\limits_{\boldsymbol{x},T,s} & s \\ \text{s.t.} & \ln P - \ln P_i^{sat} - \ln \gamma_i - s \;\leq\; 0 \quad \forall\, i \in N \\ & -\ln P + \ln P_i^{sat} + \ln \gamma_i - s \;\leq\; 0 \quad \forall\, i \in N \\ & \sum\limits_{i \in N} x_i \;=\; 1 \\ & 0 < \boldsymbol{x} < 1 \\ & s \geq 0. \end{array} \tag{24.12}$$

Note that the first two sets of constraints in (24.12) represent the nonlinear equation (24.10) for $x_i \neq 0, \forall\, i \in N$ in the form of two inequalities. For azeotropes in which less than N of the components participate (a k-ary azeotrope where $k \leq N$), the case where $x_i = 0$ for one or more components must be accounted for. This can be done by multiplying the equilibrium constraints used in (24.11) by x_i. The general search for all k-ary homogeneous azeotropes is formulated as:

$$\begin{array}{ll} \min\limits_{\boldsymbol{x},T,s} & s \\ \text{s.t.} & x_i(\ln P - \ln P_i^{sat} - \ln \gamma_i) - s \;\leq\; 0 \quad \forall\, i \in N \\ & x_i(-\ln P + \ln P_i^{sat} + \ln \gamma_i) - s \;\leq\; 0 \quad \forall\, i \in N \\ & \sum\limits_{i \in N} x_i \;=\; 1 \\ & 0 \leq \boldsymbol{x} \leq 1 \\ & s \geq 0. \end{array} \tag{24.13}$$

24.4 ACTIVITY COEFFICIENT MODELS

The problem formulations (24.12) and (24.13) contain several nonconvex functions. If a local optimization approach is used to solve these problems, it is likely that some of the multiple global minima will be missed, or even that none will be found. Therefore, in order to guarantee that *all* global minima are enclosed, it is necessary to develop a convex relaxation of the problem. A lower bound on the solution of (24.12) and (24.13) is obtained by replacing each nonconvex constraint with a tight convex underestimator. A convex relaxation is developed by partitioning each nonlinear function in the constraint equations into convex and nonconvex terms. Tight convex lower bounding functions are then developed and each nonconvex term in the constraints is replaced.

This section analyzes the equations in the azeotropy problem formulation which contain nonconvex terms. The saturated vapor pressure equation is examined first, then the Wilson equation, the NRTL equation, the UNIQUAC equation, and the UNIFAC equation are discussed.

24.4.1 Saturated Vapor Pressure Equation

The saturated vapor pressure is calculated using the Antoine equation:

$$\ln P_i^{sat} = a_i - \frac{b_i}{T + c_i} \tag{24.14}$$

where a_i, b_i, and c_i are constants. The parameters a_i and b_i are always positive, while c_i may be positive or negative but $\mid c_i \mid < T$. In the problem of finding all N-ary azeotropes, given by formulation (24.12), this term appears in the constraints as both $(-\ln P_i^{sat})$ and $(+\ln P_i^{sat})$. It can be easily shown that the term:

$$-\frac{b_i}{T+c_i} \tag{24.15}$$

is concave in T. As a result, in the first set of constraints of (24.12) which have $(-\ln P_i^{sat})$, this term is convex and no underestimation is needed. In the second set of constraints however, this term is concave and a convex underestimator is required. The convex envelope for this concave term is :

$$-\frac{b_i}{T+c_i} \geq -\frac{b_i}{T^L+c_i} + \frac{-\frac{b_i}{T^U+c_i} + \frac{b_i}{T^L+c_i}}{T^U-T^L}(T-T^L) \tag{24.16}$$

where T^L, T^U are the lower and upper bounds on the temperature in the current region.

In the formulation for finding all k-ary azeotropes (24.13), the Antoine equation is multiplied by the liquid mole fraction, x_i. In this case, both the positive and negative terms $(-x_i \ln P_i^{sat})$ and $(+x_i \ln P_i^{sat})$ are nonconvex. These terms have the form:

$$x_i \ln P_i^{sat} = f(x_i) \cdot \frac{1}{g(T)} \tag{24.17}$$

where $f(x_i)$ and $g(T)$ are univariate functions. As we discussed in Chapter 23, Maranas and Floudas (1995) extended the work of Al-Khayyal and Falk (1983) on the convex lower bound of the bilinear product xy to products and quotients of univariate functions, that is, $f(x)g(y)$ and $\frac{f(x)}{g(y)}$. For a term of the form $\frac{f(x)}{g(y)}$, where $x^L \geq 0$ and $x^U \geq 0$, the convex lower bound is given by:

$$\begin{aligned} \frac{f(x)}{g(y)} \quad \geq \quad \max \quad & \left\{ \frac{f^L(x)}{g(y)} + \frac{f(x)}{g^U(y)} - \frac{f^L(x)}{g^U(y)}, \right. \\ & \left. \frac{f^U(x)}{g(y)} + \frac{f(x)}{g^L(y)} - \frac{f^U(x)}{g^L(y)} \right\}. \end{aligned}$$

where $f^L(x)$ refers to the minimum of f over the domain $x^L \leq x \leq x^U$ and $f^U(x)$ is the maximum of f. The same notation is true for $g(y)$ also. When applied to the current problem, the convex lower bound becomes:

$$\begin{aligned} \frac{b_i x_i}{T+c_i} \quad \geq \quad \max \quad & \left\{ \frac{b_i x_i^L}{T+c_i} + \frac{b_i x_i}{T^U+c_i} - \frac{b_i x_i^L}{T^U+c_i}, \right. \\ & \left. \frac{b_i x_i^U}{T+c_i} + \frac{b_i x_i}{T^L+c_i} - \frac{b_i x_i^U}{T^L+c_i} \right\}. \end{aligned}$$

For the case when a convex underestimator for a term of the form $-\frac{f(x)}{g(y)}$ is needed a different formula must be used :

$$-\frac{f(x)}{g(y)} \geq \max \left\{ -\frac{f(x)}{g^L(y)} + \frac{f^L(x)g(y)}{g^L(y)g^U(y)} - \frac{f^L(x)}{g^U(y)}, \right. \\ \left. -\frac{f(x)}{g^U(y)} + \frac{f^U(x)g(y)}{g^L(y)g^U(y)} - \frac{f^U(x)}{g^L(y)} \right\}.$$

So the convex lower bounding function for the term of this form is:

$$-\frac{b_i x_i}{T+c_i} \geq \max \left\{ -\frac{b_i x_i}{T^L+c_i} + \frac{b_i x_i^L}{(T^L+c_i)(T^U+c_i)}T - \frac{b_i x_i^L}{T^U+c_i}, \right. \\ \left. -\frac{b_i x_i}{T^U+c_i} + \frac{b_i x_i^U}{(T^L+c_i)(T^U+c_i)}T - \frac{b_i x_i^U}{T^L+c_i} \right\}.$$

24.4.2 Wilson Equation

Wilson (1964) developed an equation to model the fluid phase excess Gibbs free energy for vapor-liquid systems. The Wilson equation is often used to model solutions containing polar and nonpolar components, and takes the form:

$$\ln \gamma_i = 1 - \ln \left(\sum_{j \in N} x_j \Lambda_{ij} \right) - \sum_{j \in N} \frac{x_j \Lambda_{ji}}{\sum_{k \in N} x_k \Lambda_{jk}} \tag{24.18}$$

where Λ_{ij} is the nonsymmetric binary interaction parameter between components i and j with $\Lambda_{ii} = 1$. When the Wilson equation is used, the problem of finding all N-ary homogeneous azeotropes is formulated as:

$$\begin{aligned} \min_{\mathbf{x},s} \quad & s \\ \text{s.t.} \quad & \ln \left(\sum_{j \in N} x_j \Lambda_{ij} \right) + \sum_{j \in N} s_j^1 \Lambda_{ji} + \tfrac{b_i}{T+c_i} \\ & + \ln P - a_i - 1 - s \leq 0 \quad \forall\, i \in N \\ & - \ln \left(\sum_{j \in N} x_j \Lambda_{ij} \right) + \sum_{j \in N} s_j^2 \Lambda_{ji} - \tfrac{b_i}{T+c_i} \\ & - \ln P + a_i + 1 - s \leq 0 \quad \forall\, i \in N \\ & \sum_{i \in N} x_i = 1 \\ & 0 < \mathbf{x} < 1 \\ & s \geq 0 \end{aligned} \tag{24.19}$$

where,

$$s_j^1 = \frac{x_j}{\sum_{k \in N} x_k \Lambda_{jk}} \tag{24.20}$$

$$s_j^2 = -\frac{x_j}{\sum\limits_{k \in N} x_k \Lambda_{jk}}. \tag{24.21}$$

The s_j^1 and s_j^2 equations are used to isolate the nonconvex terms in the complicated summation expression. The Wilson expression for the activity coefficients contains three different nonconvex terms. The first nonconvex term is $(\ln \sum\limits_{j \in N} x_j \Lambda_{ij})$. The second and third nonconvex terms are s_j^1 and s_j^2. As was described in the preceding section on the Antoine equation, multiplication of the equilibrium constraints by x_i in the problem of finding all k-ary azeotropes further complicates the nonconvex expressions. For the k-ary azeotrope problem (24.13), the Wilson equation now contains the following nonconvex terms:

$$x_i \ln(\sum_{j \in N} x_j \Lambda_{ij}), \quad -x_i \ln(\sum_{j \in N} x_j \Lambda_{ij}), \quad x_i \sum_{j \in N} s_j^1 \Lambda_{ji}, \quad x_i \sum_{j \in N} s_j^2 \Lambda_{ji}. \tag{24.22}$$

A convex lower bounding function must be developed for each of these nonconvex terms. The procedure for constructing the underestimating functions for the Wilson equation is presented next.

Convex Underestimators for the Wilson Equation. The first nonconvex term in the problem of finding all N-ary azeotropes using the Wilson formulation that will be examined is $\ln \left\{ \sum\limits_{j \in N} x_j \Lambda_{ij} \right\}$, which is concave in x_i, for $i = 1, 2, ..., N$. Therefore, this term can be underestimated by the following expression:

$$\ln \sum_{j \in N} x_j \Lambda_{ij} \geq \ln m_i^L + \frac{\ln \frac{m_i^U}{m_i^L}}{(m_i^U - m_i^L)} (\sum_{j \in N} x_j \Lambda_{ij} - m_i^L) \tag{24.23}$$

where,

$$\begin{aligned} m_i^L \;=\; \min \quad & \sum_{j \in N} x_j \Lambda_{ij} \\ \text{s.t.} \quad & \sum_{j \in N} x_j = 1 \\ & x_j^L \leq x_j \leq x_j^U \end{aligned} \tag{24.24}$$

and,

$$\begin{aligned} m_i^U \;=\; \max \quad & \sum_{j \in N} x_j \Lambda_{ij} \\ \text{s.t.} \quad & \sum_{j \in N} x_j = 1 \\ & x_j^L \leq x_j \leq x_j^U. \end{aligned} \tag{24.25}$$

The second type of nonconvex terms found in the Wilson formulation have the form:

$$\left(s_j^1 = +\frac{x_j}{\sum\limits_{k \in N} x_k \Lambda_{jk}} \right) \text{ and } \left(s_j^2 = -\frac{x_j}{\sum\limits_{k \in N} x_k \Lambda_{jk}} \right).$$

Following the method presented in Chapter 23 and in Maranas and Floudas (1995) for products of univariate functions, the convex underestimators for these terms are:

$$\begin{aligned} s_j^1 \;\geq\; & \max \left\{ \frac{x_j^L}{\sum\limits_{k \in N} x_k \Lambda_{jk}} + \frac{x_j}{m_j^U} - \frac{x_j^L}{m_j^U}, \right. \\ & \left. \frac{x_j^U}{\sum\limits_{k \in N} x_k \Lambda_{jk}} + \frac{x_j}{m_j^L} - \frac{x_j^U}{m_j^L} \right\}. \\ s_j^2 \;\geq\; & \max \left\{ -\frac{x_j}{m_j^L} + \frac{x_j^L \sum\limits_{k \in N} x_k \Lambda_{jk}}{m_j^L m_j^U} - \frac{x_j^L}{m_j^U}, \right. \\ & \left. -\frac{x_j}{m_j^U} + \frac{x_j^U \sum\limits_{k \in N} x_k \Lambda_{jk}}{m_j^L m_j^U} - \frac{x_j^U}{m_j^L} \right\}. \end{aligned}$$

For the k-ary azeotrope problem (24.13) the nonconvex terms are:

$$\left(x_i \ln(\sum_{j \in N} x_j \Lambda_{ij}) \right), \left(-x_i \ln(\sum_{j \in N} x_j \Lambda_{ij}) \right),$$
$$\left(x_i \sum_{j \in N} s_j^1 \Lambda_{ji} \right), \left(-x_i \sum_{j \in N} s_j^1 \Lambda_{ji} \right).$$

The first nonconvex term above can be considered as the product of two univariate functions, $x_i u_i$, where u_i is the log term in the equation. The convex underestimator for this product is (Al-Khayyal and Falk, 1983):

$$\begin{aligned} x_i u_i \;\geq\; & \max \; \{ x_i^L u_i + x_i u_i^L - x_i^L u_i^L, \\ & x_i^U u_i + x_i u_i^U - x_i^U u_i^U \}. \end{aligned}$$

where,

$$u_i = \ln(\sum_{j \in N} x_j \Lambda_{ij})$$

and,

$$\begin{aligned} u_i^L &= \ln(\min \sum_{j\in N} x_j\Lambda_{ij}) \\ &\quad \text{s.t.} \sum_{j\in N} x_j = 1 \\ &\quad x_j^L \le x_j \le x_j^U \\ u_i^U &= \ln(\max \sum_{j\in N})x_j\Lambda_{ij}) \\ &\quad \text{s.t.} \sum_{j\in N} x_j = 1 \\ &\quad x_j^L \le x_j \le x_j^U . \end{aligned}$$

Note that $x_i^L u_i$ and $x_i^U u_i$ are nonconvex, so the convex underestimator should be used in place of u_i.

Similarly, the second nonconvex term is of the form $x_i(-u_i)$. The convex underestimator for this product is:

$$\begin{aligned} -x_i u_i \quad \ge \quad \max \ & \{-x_i^U u_i - x_i u_i^L + x_i^U u_i^L, \\ & -x_i^L u_i - x_i u_i^U + x_i^L u_i^U\} , \end{aligned}$$

and the u_i term is nonconvex and must be replaced by the convex underestimator. Consider the third and fourth nonconvex terms to be of the form:

$$t_i = x_i(\sum_{j\in N} u_j)$$

which is a sum of products of univariate functions. Each term can be underestimated by:

$$\begin{aligned} x_i u_j \quad \ge \quad \max \ & \{x_i^L u_j + x_i u_j^L - x_i^L u_j^L, \\ & x_i^U u_j + x_i u_j^U - x_i^U u_j^U\} . \end{aligned}$$

Now when these terms are summed over each j in the set of components N, the following formula results:

$$\begin{aligned} \sum_{j\in N} x_i u_j \quad \ge \quad \max \ & \Bigg\{ x_i^L \sum_{j\in N} u_j + x_i \sum_{j\in N} u_j^L - x_i^L \sum_{j\in N} u_j^L, \\ & x_i^U \sum_{j\in N} u_j + x_i \sum_{j\in N} u_j^U - x_i^U \sum_{j\in N} u_j^U \Bigg\} . \end{aligned}$$

However, the term u_j is nonconvex, so it must be replaced by a valid convex underestimator. Note that $u_j = s_j^1\Lambda_{ij}$ and a convex underestimator for this term was developed earlier.

The upper and lower bounds for the summation terms can be obtained essentially by minimizing the numerator of each term while maximizing the denominator. Substituting in for s_j^1 gives:

$$\begin{aligned}\sum_{j\in N} u_j^L &= \sum_{j\in N} \Lambda_{ji} \frac{\min\limits_{j\in N} x_j}{\max\limits_{k\in N} x_k \Lambda_{jk}} \\ &= \sum_{j\in N} \frac{x_j^L \Lambda_{ji}}{m_j^U}.\end{aligned}$$

The term m_j^U was defined in Section 24.4 by Equation (24.25). The term $\sum\limits_{j\in N} u_i^U$ is determined in a similar manner and is:

$$\sum_{j\in N} u_j^U = \sum_{j\in N} \frac{x_j^U \Lambda_{ji}}{m_j^L}.$$

Therefore, the nonconvex term:

$$t_i^1 = x_i \sum_{j\in N} s_j^1 \Lambda_{ji}$$

is underestimated by:

$$\begin{aligned} t_i^1 \geq \max \Bigg\{ & x_i^L \sum_{j\in N} u_j + x_i \sum_{j\in N} \frac{x_j^L \Lambda_{ji}}{m_j^U} - x_i^L \sum_{j\in N} \frac{x_j^L \Lambda_{ji}}{m_j^U}, \\ & x_i^U \sum_{j\in N} u_j + x_i \sum_{j\in N} \frac{x_j^U \Lambda_{ji}}{m_j^L} - x_i^U \sum_{j\in N} \frac{x_j^U \Lambda_{ji}}{m_j^L} \Bigg\}. \end{aligned}$$

where

$$\begin{aligned} u_j \geq \max \Bigg\{ & \frac{x_j^L \Lambda_{ji}}{\sum\limits_{k\in N} x_k \Lambda_{jk}} + \frac{x_j \Lambda_{ji}}{m_j^U} - \frac{x_j^L \Lambda_{ji}}{m_j^U}, \\ & \frac{x_j^U \Lambda_{ji}}{\sum\limits_{k\in N} x_k \Lambda_{jk}} + \frac{x_j \Lambda_{ji}}{m_j^L} - \frac{x_j^U \Lambda_{ji}}{m_j^L} \Bigg\}. \end{aligned}$$

Through a similar analysis, the term:

$$t_i^2 = -x_i \sum_{j\in N} s_j^1 \Lambda_{ji}$$

can be underestimated by:

$$t_i^2 \geq \max \left\{ x_i^L \sum_{j\in N} v_j - x_i \sum_{j\in N} \frac{x_j^L \Lambda_{ji}}{m_j^U} + x_i^L \sum_{j\in N} \frac{x_j^L \Lambda_{ji}}{m_j^U}, \right.$$
$$\left. x_i^U \sum_{j\in N} v_j - x_i \sum_{j\in N} \frac{x_j^U \Lambda_{ji}}{m_j^L} + x_i^U \sum_{j\in N} \frac{x_j^U \Lambda_{ji}}{m_j^L} \right\}.$$

where

$$v_j \geq \max \left\{ -\frac{x_j}{m_j^L} + \frac{x_j^L \sum_{k\in N} x_k \Lambda_{jk}}{m_j^L m_j^U} - \frac{x_j^L}{m_j^U}, \right.$$
$$\left. -\frac{x_j}{m_j^U} + \frac{x_j^U \sum_{k\in N} x_k \Lambda_{jk}}{m_j^L m_j^U} - \frac{x_j^U}{m_j^L} \right\}.$$

24.4.3 NRTL Equation

The NRTL equation was derived by Renon and Prausnitz (1968) for fluid phase activity coefficients using the Non-Randomness assumption and Scott's Two-Liquid theory. The NRTL formula for the activity coefficient is:

$$\ln \gamma_i = \frac{\sum_{j\in N} \tau_{ji} G_{ji} x_j}{\sum_{j\in N} G_{ji} x_j} + \sum_{j\in N} \frac{G_{ij} x_j}{\sum_{k\in N} G_{kj} x_k} \left(\tau_{ij} - \frac{\sum_{k\in N} \tau_{kj} G_{kj} x_k}{\sum_{k\in N} G_{kj} x_k}\right). \tag{24.26}$$

Substituting this expression into formulation (24.12) yields:

$$\begin{aligned}
\min_{\boldsymbol{x},s} \quad & s \\
\text{s.t.} \; & -\sum_{j\in N} \tau_{ij} \frac{G_{ij} x_j}{y_j} + \sum_{j\in N} \frac{G_{ij} x_j}{y_j} \frac{w_j}{y_j} - \frac{w_i}{y_i} + \frac{b_i}{T+c_i} \\
& + \ln P - a_i - s \leq 0 \quad \forall\, i \in N \\
& \sum_{j\in N} \tau_{ij} \frac{G_{ij} x_j}{y_j} - \sum_{j\in N} \frac{G_{ij} x_j}{y_j} \frac{w_j}{y_j} + \frac{w_i}{y_i} - \frac{b_i}{T+c_i} \\
& - \ln P + a_i - s \leq 0 \quad \forall\, i \in N \\
& \sum_{i\in N} x_i = 1 \\
& 0 < \boldsymbol{x} < 1 \\
& s \geq 0
\end{aligned} \tag{24.27}$$

where $\boldsymbol{x}$ denotes the vector of compositions, and

$$\begin{aligned}
y_i &= \sum_{j\in N} G_{ji} x_j \\
w_i &= \sum_{j\in N} \tau_{ji} G_{ji} x_j.
\end{aligned} \tag{24.28}$$

The interaction parameters, τ_{ij} and G_{ij}, are functions of temperature. However, for most systems, the temperature dependence of the Antoine equation for the vapor pressure outweighs the temperature dependence of the activity coefficient equation. Therefore, in this analysis, the interaction parameters are considered to be constants, and are calculated using a reference temperature located between the upper and lower bounds for the system being modeled. Taking this into account, formulation (24.27) contains nonconvex terms of the form:

$$u_i = \frac{f_i}{g_i}$$

and,

$$v_i = \frac{f_i^1 f_i^2}{g_i^1 g_i^2}.$$

The task of constructing convex underestimators for nonconvex terms of the second form is a difficult one. As discussed in Chapter 23 and in the work of Maranas and Floudas (1995), products of N univariate functions can be underestimated by successively convex lower bounding pairs of univariate functions in a recursive manner until no pairs are left. This procedure is applied in the next section to the above nonconvex terms and convex underestimators are derived explicitly.

Convex Underestimating Functions for the NRTL Equation. In the NRTL equation formulation for finding all N-ary azeotropes, (24.27), three different types of nonconvex terms are encountered:

$$\begin{aligned}
+p_i &= \frac{w_i}{y_i} \\
-p_i &= -\frac{w_i}{y_i} \\
+q_i &= \sum_{j \in N} \tau_{ij} G_{ij} \frac{x_j}{y_j} \\
-q_i &= -\sum_{j \in N} \tau_{ij} G_{ij} \frac{x_j}{y_j} \\
+r_i &= \sum_{j \in N} G_{ij} \frac{x_j w_j}{y_j y_j} \\
-r_i &= -\sum_{j \in N} G_{ij} \frac{x_j w_j}{y_j y_j}
\end{aligned}$$

where

$$\begin{aligned}
y_i &= \sum_{j \in N} G_{ji} x_j \\
w_i &= \sum_{j \in N} \tau_{ji} G_{ji} x_j
\end{aligned}$$

so $x_i, y_i \geq 0$, but w_i can be either positive or negative.

Following the method of Maranas and Floudas (1995) for fractional terms, the first two nonconvex terms can be underestimated by:

$$\begin{aligned}
\frac{w_i}{y_i} &\geq \max \left\{ \begin{bmatrix} \frac{w_i^L}{y_i} + \frac{w_i}{y_i^U} - \frac{w_i^L}{y_i^U} & \text{if } w_i^L \geq 0 \\ \frac{w_i}{y_i^U} - \frac{w_i^L y_i}{y_i^L y_i^U} + \frac{w_i^L}{y_i^L} & \text{if } w_i^L < 0 \end{bmatrix} \right. \\
&\qquad \left. \begin{bmatrix} \frac{w_i^U}{y_i} + \frac{w_i}{y_i^L} - \frac{w_i^U}{y_i^L} & \text{if } w_i^U \geq 0 \\ \frac{w_i}{y_i^L} - \frac{w_i^U y_i}{y_i^L y_i^U} + \frac{w_i^U}{y_i^U} & \text{if } w_i^U < 0 \end{bmatrix} \right\} \\
-\frac{w_i}{y_i} &\geq \max \left\{ \begin{bmatrix} -\frac{w_i^L}{y_i} - \frac{w_i}{y_i^L} + \frac{w_i^L}{y_i^L} & \text{if } w_i^L \leq 0 \\ -\frac{w_i}{y_i^L} + \frac{w_i^L y_i}{y_i^L y_i^U} - \frac{w_i^L}{y_i^U} & \text{if } w_i^L > 0 \end{bmatrix} \right. \\
&\qquad \left. \begin{bmatrix} -\frac{w_i^U}{y_i} - \frac{w_i}{y_i^U} + \frac{w_i^U}{y_i^U} & \text{if } w_i^U \leq 0 \\ -\frac{w_i}{y_i^U} + \frac{w_i^U y_i}{y_i^L y_i^U} - \frac{w_i^U}{y_i^L} & \text{if } w_i^U > 0 \end{bmatrix} \right\}
\end{aligned}$$

where

$$\begin{aligned}
w_j^L &= \min_{k \in N} \tau_{kj} G_{kj} x_k \\
w_j^L &= \max_{k \in N} \tau_{kj} G_{kj} x_k \\
y_j^L &= \min_{k \in N} G_{kj} x_k \\
y_j^U &= \max_{k \in N} G_{kj} x_k
\end{aligned}$$

In the second type of nonconvex term, each fractional term in the summation can be underestimated by:

$$\begin{aligned}
\frac{x_j}{y_j} &\geq \max \left\{ \frac{x_j^L}{y_j} + \frac{x_j}{y_j^U} - \frac{x_j^L}{y_j^U}, \right. \\
&\qquad \left. \frac{x_j^U}{y_j} + \frac{x_j}{y_j^L} - \frac{x_j^U}{y_j^L} \right\} \\
-\frac{x_j}{y_j} &\geq \max \left\{ -\frac{x_j}{y_j^L} + \frac{x_j^L y_j}{y_j^L y_j^U} - \frac{x_j^L}{y_j^U}, \right. \\
&\qquad \left. -\frac{x_j}{y_j^U} + \frac{x_j^U y_j}{y_j^L y_j^U} - \frac{x_j^U}{y_j^L} \right\}.
\end{aligned}$$

Therefore, since $\sum_j (\max x_j) = \max\left(\sum_j x_j\right)$, the convex underestimator for the second term is:

$$\begin{aligned}
\sum_{j\in N} \tau_{ij} G_{ij} \frac{x_j}{y_j} \quad &\geq \quad \max \left\{ \sum_{j\in N} \tau_{ij} G_{ij} \frac{x_j^L}{y_j} + \sum_{j\in N} \tau_{ij} G_{ij} \frac{x_j}{y_j^U} - \sum_{j\in N} \tau_{ij} G_{ij} \frac{x_j^L}{y_j^U}, \right. \\
&\qquad \left. \sum_{j\in N} \tau_{ij} G_{ij} \frac{x_j^U}{y_j} + \sum_{j\in N} \tau_{ij} G_{ij} \frac{x_j}{y_j^L} - \sum_{j\in N} \tau_{ij} G_{ij} \frac{x_j^U}{y_j^L} \right\} \\
-\sum_{j\in N} \tau_{ij} G_{ij} \frac{x_j}{y_j} \quad &\geq \quad \max \left\{ -\sum_{j\in N} \tau_{ij} G_{ij} \frac{x_j}{y_j^L} + \sum_{j\in N} \tau_{ij} G_{ij} \frac{x_j^L y_j}{y_j^L y_j^U} - \sum_{j\in N} \tau_{ij} G_{ij} \frac{x_j^L}{y_j^U}, \right. \\
&\qquad \left. -\sum_{j\in N} \tau_{ij} G_{ij} \frac{x_j}{y_j^U} + \sum_{j\in N} \tau_{ij} G_{ij} \frac{x_j^U y_j}{y_j^L y_j^U} - \sum_{j\in N} \tau_{ij} G_{ij} \frac{x_j^U}{y_j^L} \right\}.
\end{aligned}$$

The third term is more difficult to underestimate. A convex underestimator can be constructed by successively finding the convex lower bounding function for pairs of univariate functions in a recursive manner. The recursive procedure for constructing convex underestimators from pairs of univariate functions, as developed by Maranas and Floudas (1995), is illustrated below.

First, the fractional term $\left(\frac{x_j}{y_j}\right)$ is considered. Since x_j^L and x_j^U are both always greater than zero, this term can be underestimated by:

$$\begin{aligned}
\frac{x_j}{y_j} \geq \lambda_j^2 \quad = \quad \max \quad &\left\{ \frac{x_j^L}{y_j} + \frac{x_j}{y_j^U} - \frac{x_j^L}{y_j^U}, \right. \\
&\left. \frac{x_j^U}{y_j} + \frac{x_j}{y_j^L} - \frac{x_j^U}{y_j^L} \right\}
\end{aligned}$$

and since $(x_j - x_j^L)$ is always positive, λ_j^2 is always positive. Next, a convex underestimating function is constructed for the term $\left(\frac{\lambda_j^2}{y_j}\right)$:

$$\begin{aligned}
\frac{\lambda_j^2}{y_j} \geq \lambda_j^1 \quad = \quad \max \quad &\left\{ \frac{\lambda_j^{2,L}}{y_j} + \frac{\lambda_j^2}{y_j^U} - \frac{\lambda_j^{2,L}}{y_j^U}, \right. \\
&\left. \frac{\lambda_j^{2,U}}{y_j} + \frac{\lambda_j^2}{y_j^L} - \frac{\lambda_j^{2,U}}{y_j^L} \right\}
\end{aligned}$$

where $\left(\lambda_j^{2,L} = \frac{x_j^L}{y_j^U}\right)$ and $\left(\lambda_j^{2,U} = \frac{x_j^U}{y_j^L}\right)$.

Finally, convex underestimating functions for the terms $(\lambda_j^1 w_j)$ and $(\lambda_j^1(-w_j))$ are constructed:

$$\begin{aligned}
\lambda_j^1 w_j &\geq \max \left\{ w_j^L \lambda_j^1 + w_j \lambda_j^{1,L} - w_j^L \lambda_j^{1,L}, \right. \\
&\qquad \left. w_j^U \lambda_j^1 + w_j \lambda_j^{1,U} - w_j^U \lambda_j^{1,U} \right\} \\
\lambda_j^1(-w_j) &\geq \max \left\{ -w_j^U \lambda_j^1 - w_j \lambda_j^{1,L} + w_j^U \lambda_j^{1,L}, \right. \\
&\qquad \left. -w_j^L \lambda_j^1 - w_j \lambda_j^{1,U} + w_j^L \lambda_j^{1,U} \right\}
\end{aligned}$$

where $\left(\lambda_j^{1,L} = \frac{\lambda_j^{2,L}}{y_j^U} = \frac{x_j^L}{y_j^U y_j^U}\right)$ and $\left(\lambda_j^{1,U} = \frac{\lambda_j^{2,U}}{y_j^L} = \frac{x_j^U}{y_j^L y_j^L}\right)$. Combining these terms gives a convex underestimating function for the each fractional term in the summation.

$$\begin{aligned}
\frac{x_j w_j}{y_j y_j} &\geq \max \left\{ w_j^L \lambda_j^1 + \frac{w_j x_j^L}{y_j^U y_j^U} - \frac{w_j^L x_j^L}{y_j^U y_j^U}, \right. \\
&\qquad \left. w_j^U \lambda_j^1 + \frac{w_j x_j^U}{y_j^L y_j^L} - \frac{w_j^U x_j^U}{y_j^L y_j^L} \right\} \\
-\frac{x_j w_j}{y_j y_j} &\geq \max \left\{ -w_j^U \lambda_j^1 - \frac{w_j x_j^L}{y_j^U y_j^U} + \frac{w_j^U x_j^L}{y_j^U y_j^U}, \right. \\
&\qquad \left. -w_j^L \lambda_j^1 - \frac{w_j x_j^U}{y_j^L y_j^L} + \frac{w_j^L x_j^U}{y_j^L y_j^L} \right\}
\end{aligned}$$

and λ_j^1, λ_j^2, y_j^L, y_j^U, w_j^L, and w_j^U are as defined above. Now, by the same argument as was given for the second type of nonconvex term, the convex underestimator for the third type of term is:

$$\begin{aligned}
\sum_{j \in N} G_{ij} \frac{x_j w_j}{y_j y_j} &\geq \max \left\{ \sum_{j \in N} G_{ij} w_j^L \lambda_j^1 + \sum_{j \in N} G_{ij} \frac{w_j x_j^L}{y_j^U y_j^U} - \sum_{j \in N} G_{ij} \frac{w_j^L x_j^L}{y_j^U y_j^U}, \right. \\
&\qquad \left. \sum_{j \in N} G_{ij} w_j^U \lambda_j^1 + \sum_{j \in N} G_{ij} \frac{w_j x_j^U}{y_j^L y_j^L} - \sum_{j \in N} G_{ij} \frac{w_j^U x_j^U}{y_j^L y_j^L} \right\} \\
-\sum_{j \in N} G_{ij} \frac{x_j w_j}{y_j y_j} &\geq \max \left\{ -\sum_{j \in N} G_{ij} w_j^U \lambda_j^1 - \sum_{j \in N} G_{ij} \frac{w_j x_j^L}{y_j^U y_j^U} + \sum_{j \in N} G_{ij} \frac{w_j^U x_j^L}{y_j^U y_j^U}, \right. \\
&\qquad \left. -\sum_{j \in N} G_{ij} w_j^L \lambda_j^1 - \sum_{j \in N} G_{ij} \frac{w_j x_j^U}{y_j^L y_j^L} + \sum_{j \in N} G_{ij} \frac{w_j^L x_j^U}{y_j^L y_j^L} \right\}.
\end{aligned}$$

24.4.4 *UNIQUAC Equation*

The UNIQUAC equation was originally developed by Abrams and Prausnitz (1975). They postulated that the excess Gibbs energy depended on two effects: a combinatorial contribution due to the differences in the sizes and shapes of the components, and a residual contribution due to the energetic interactions between the components. The original equation was modified by Anderson and Prausnitz (1978) in order to improve predictions for systems containing polar molecules such as water and alcohols. The activity coefficient for the UNIQUAC equation is given as:

$$\ln \gamma_i = \ln \gamma_i^C + \ln \gamma_i^R \tag{24.29}$$

where,

$$\ln \gamma_i^C = \ln \frac{\phi_i}{x_i} + \frac{z}{2} q_i \ln \frac{\theta_i}{\phi_i} + l_i - \frac{\phi_i}{x_i} \sum_{j \in N} l_j x_j \tag{24.30}$$

and,

$$\ln \gamma_i^R = q_i'(1 - \ln(\sum_{j \in N} \tau_{ji} \theta_j') - \sum_{j \in N} \frac{\tau_{ij} \theta_j'}{\sum_{k \in N} \theta_k' \tau_{kj}}). \tag{24.31}$$

The combinatorial and residual contributions to the activity coefficient are represented by γ_i^C and γ_i^R respectively. The τ_{ij} are non-symmetric binary interaction parameters, q_i, q_i', and r_i are structural parameters of the pure components, and z is a lattice coordination number. The values l_i, θ_i, θ_i', and ϕ_i are defined in terms of the parameters described above:

$$\begin{aligned} l_i &= \tfrac{z}{2}(r_i - q_i) - (r_i - 1) && \forall\, i \in N \\ \theta_i &= \frac{q_i x_i}{\sum_{j \in N} q_j x_j} && \forall\, i \in N \\ \theta_i' &= \frac{q_i' x_i}{\sum_{j \in N} q_j' x_j} && \forall\, i \in N \\ \phi_i &= \frac{r_i x_i}{\sum_{j \in N} r_j x_j} && \forall\, i \in N. \end{aligned} \tag{24.32}$$

When the definitions of these parameters are substituted into the $\ln \gamma_i^C$ and $\ln \gamma_i^R$ expressions, and these two are added together to form the complete $\ln \gamma_i$ equation some simplifications can be made. The final expression is:

$$\begin{aligned} \ln \gamma_i \;=\; & (\frac{z}{2} q_i - 1) \ln(\sum_{j \in N} r_j x_j) - \frac{z}{2} q_i \ln(\sum_{j \in N} q_j x_j) - r_i \frac{\sum_{j \in N} l_j x_j}{\sum_{j \in N} r_j x_j} \\ & - q_i' \ln(\sum_{j \in N} \tau_{ji} q_j' x_j) + q_i' \ln(\sum_{j \in N} q_j' x_j) - q_i' \sum_{j \in N} \frac{\tau_{ij} q_j' x_j}{\sum_{k \in N} \tau_{kj} q_k' x_k} \\ & + \left(\ln r_i + \frac{z}{2} q_i \ln(\frac{q_i}{r_i}) + l_i + q_i' \right). \end{aligned}$$

As is evident, this expression contains many nonconvex terms. The first four nonconvex terms are log terms:

$$\ln\left(\sum_{j\in N} r_j x_j\right), \quad \ln\left(\sum_{j\in N} q_j x_j\right), \quad \ln\left(\sum_{j\in N} q'_j x_j\right), \quad \text{and} \quad \ln\left(\sum_{j\in N} \tau_{ji} q'_j x_j\right). \tag{24.33}$$

The last four nonconvex terms are fractional terms:

$$+\frac{\sum_{j\in N} l_j x_j}{\sum_{l\in N} r_l x_l}, \quad -\frac{\sum_{j\in N} l_j x_j}{\sum_{l\in N} r_l x_l}, \quad +\frac{\sum_{j\in N} \tau_{ij} q'_j x_j}{\sum_{l\in N} \tau_{lj} q'_l x_l}, \quad \text{and} \quad -\frac{\sum_{j\in N} \tau_{ij} q'_j x_j}{\sum_{l\in N} \tau_{lj} q'_l x_l}. \tag{24.34}$$

Note that the binary interaction parameter, τ_{ij}, is considered to be a constant in this analysis, as described in Section 24.4.3. The global optimization problem of finding all N-ary homogeneous azeotropes using the UNIQUAC model is formulated as follows:

$$\begin{aligned}
\min_{\boldsymbol{x},s} \quad & s \\
\text{s.t.} \quad & (\tfrac{z}{2}q_i - 1)\ln(\sum_{j\in N} r_j x_j) - \tfrac{z}{2}q_i \ln(\sum_{j\in N} q_j x_j) - r_i \frac{\sum_{j\in N} l_j x_j}{\sum_{j\in N} r_j x_j} \\
& -q'_i \ln(\sum_{j\in N} \tau_{ji} q'_j x_j) + q'_i \ln(\sum_{j\in N} q'_j x_j) - q'_i \sum_{j\in N} \frac{\tau_{ij} q'_j x_j}{\sum_{k\in N} \tau_{kj} q'_k x_k} \\
& + \left(\ln r_i + \tfrac{z}{2}q_i \ln(\tfrac{q_i}{r_i}) + l_i + q'_i\right) + \tfrac{b_i}{T+c_i} + \ln P - a_i - s \;\le\; 0 \quad \forall\, i \in N \\
& -(\tfrac{z}{2}q_i - 1)\ln(\sum_{j\in N} r_j x_j) + \tfrac{z}{2}q_i \ln(\sum_{j\in N} q_j x_j) + r_i \frac{\sum_{j\in N} l_j x_j}{\sum_{j\in N} r_j x_j} \\
& +q'_i \ln(\sum_{j\in N} \tau_{ji} q'_j x_j) - q'_i \ln(\sum_{j\in N} q'_j x_j) + q'_i \sum_{j\in N} \frac{\tau_{ij} q'_j x_j}{\sum_{k\in N} \tau_{kj} q'_k x_k} \\
& - \left(\ln r_i + \tfrac{z}{2}q_i \ln(\tfrac{q_i}{r_i}) + l_i + q'_i\right) - \tfrac{b_i}{T+c_i} - \ln P + a_i - s \;\le\; 0 \quad \forall\, i \in N \\
& \sum_{i\in N} x_i \;=\; 1 \\
& 0 < \boldsymbol{x} < 1 \\
& s \ge 0.
\end{aligned} \tag{24.35}$$

Convex Underestimating Functions for the UNIQUAC Equation. The N-ary azeotrope problem using the UNIQUAC equation contains many nonlinear terms. The nonconvex terms can be grouped into two basic categories. The first group has the form:

$$\ln\left(\sum_{j\in N} c_j x_j\right).$$

The nonconvex terms in this group are:

$$\ln\left(\sum_{j\in N} r_j x_j\right), \quad \ln\left(\sum_{j\in N} q_j x_j\right), \quad \ln\left(\sum_{j\in N} q'_j x_j\right), \quad \text{and} \quad \ln\left(\sum_{j\in N} \tau_{ji} q'_j x_j\right).$$

These are all concave terms, thus convex underestimating functions can easily be constructed. When terms of this form appear as $(-\ln \sum_{j\in N} c_j x_j)$ they are convex and do not need underestimators. The convex underestimators for the first three terms above are written as:

$$\ln\left(\sum_{j\in N} c_j x_j\right) \geq \ln m^L + \frac{\ln\left\{\frac{m^U}{m^L}\right\}}{(m^U - m^L)}\left(\sum_{j\in N} c_j x_j - m^L\right)$$

for $c_j = r_j, q_j$, or q'_j and where,

$$\begin{aligned} m^U = \max \quad & \sum_{j\in N} c_j x_j \\ \text{s.t.} \quad & \sum_{j\in N} x_j = 1 \\ & x_j^L \leq x_j \leq x_j^U \end{aligned}$$

and,

$$\begin{aligned} m^L = \min \quad & \sum_{j\in N} c_j x_j \\ \text{s.t.} \quad & \sum_{j\in N} x_j = 1 \\ & x_j^L \leq x_j \leq x_j^U. \end{aligned}$$

The convex underestimator for the fourth term is very similar:

$$\ln\left(\sum_{j\in N} c_{ij} x_j\right) \geq \ln m_i^L + \frac{\ln\left\{\frac{m_i^U}{m_i^L}\right\}}{(m_i^U - m_i^L)}\left(\sum_{j\in N} c_{ij} x_j - m_i^L\right)$$

for $c_{ij} = \tau_{ji} q'_j$ and where,

$$\begin{aligned} m_i^U = \max \quad & \sum_{j\in N} \tau_{ji} q'_j x_j \\ \text{s.t.} \quad & \sum_{j\in N} x_j = 1 \\ & x_j^L \leq x_j \leq x_j^U \end{aligned}$$

and,

$$\begin{aligned} m_i^L = \min \quad & \sum_{j\in N} \tau_{ji} q'_j x_j \\ \text{s.t.} \quad & \sum_{j\in N} x_j = 1 \\ & x_j^L \leq x_j \leq x_j^U. \end{aligned}$$

The second group of nonconvex terms are fractional terms of the form:

$$c \cdot \frac{x_j}{\sum\limits_{l \in N} d_l x_l}. \tag{24.36}$$

This group includes the terms:

$$+\frac{\sum\limits_{j \in N} l_j x_j}{\sum\limits_{l \in N} r_l x_l}, \quad -\frac{\sum\limits_{j \in N} l_j x_j}{\sum\limits_{l \in N} r_l x_l}, \quad +\frac{\sum\limits_{j \in N} \tau_{ij} q'_j x_j}{\sum\limits_{l \in N} \tau_{lj} q'_l x_l}, \text{ and } -\frac{\sum\limits_{j \in N} \tau_{ij} q'_j x_j}{\sum\limits_{l \in N} \tau_{lj} q'_l x_l}.$$

For terms of this type, the sign of the constant in the numerator determines the formula that is used for the underestimating function. For the UNIQUAC equation, the parameters q_j, q'_j, r_j, and τ_{ij} are always positive, while l_j may be either positive or negative. Therefore the constant $c = +\tau_{ij} q'_j$ will always be positive and $c = -\tau_{ij} q'_j$ will always be negative. However, the sign of the constant for the first two terms, $c = +l_j$ and $c = -l_j$ must be determined before the underestimating function can be constructed. When c is positive, the underestimating function for a linear fractional term is given by:

$$\frac{c x_j}{\sum\limits_{j \in N} d_j x_j} \geq \max \left\{ \frac{c x_j^L}{\sum\limits_{l \in N} d_l x_l} + \frac{c x_j}{m^U} - \frac{c x_j^L}{m^U}, \right.$$
$$\left. \frac{c x_j^U}{\sum\limits_{l \in N} d_l x_l} + \frac{c x_j}{m^L} - \frac{c x_j^U}{m^L} \right\}.$$

When c is negative, the underestimating function is given by:

$$\frac{c x_j}{\sum\limits_{j \in N} d_j x_j} \geq \max \left\{ -\frac{c x_j}{m^L} + \frac{c x_j^L \cdot \sum\limits_{l \in N} d_l x_l}{m^L m^U} - \frac{c x_j^L}{m^U}, \right.$$
$$\left. -\frac{c x_j}{m^U} + \frac{c x_j^U \sum\limits_{l \in N} d_l x_l}{m^L m^U} - \frac{c x_j^U}{m^L} \right\}$$

where,

$$\begin{aligned} m^U = \max \quad & \sum_{l \in N} d_l x_l \\ \text{s.t.} \quad & \sum_{l \in N} x_l = 1 \\ & x_l^L \leq x_l \leq x_l^U \end{aligned}$$

and,

$$\begin{aligned} m^L = \min \quad & \sum_{l \in N} d_l x_l \\ \text{s.t.} \quad & \sum_{l \in N} x_l = 1 \\ & x_l^L \leq x_l \leq x_l^U. \end{aligned}$$

24.4.5 UNIFAC Equation

In some cases, one would like to predict the azeotropes for a system where there are insufficient equilibrium data to obtain accurate binary interaction parameter values. As a result, several solution-of-groups methods have been developed to allow equilibrium predictions in these systems. Solution-of-groups methods consider the molecules in the chemical system as groupings of elements. For example, 1-propanol contains one CH_3 group, one CH_2 group, and one CH_3OH group. The interactions between the molecules can then be approximated by determining the interactions between the groups that make up the molecules.

The most widely used solution-of-groups method is the UNIFAC equation, originally developed by Fredenslund *et al.* (1975), for which the group-interaction parameters of many pairs of groups have been calculated and compiled (Gmehling *et al.*, 1990). As its name indicates, the UNIFAC (UNIQUAC Functional group Activity Coefficients) method is based on the UNIQUAC equation. Like the UNIQUAC equation, the natural logarithm of the UNIFAC activity coefficient is the sum of a combinatorial contribution and a residual contribution:

$$\ln \gamma_i = \ln \gamma_i^C + \ln \gamma_i^R. \tag{24.37}$$

The combinatorial part, $\ln \gamma_i^C$, uses only pure-component data, and therefore no modifications of the UNIQUAC combinatorial part, (24.30), are needed. However, in the UNIFAC method, the residual part is calculated as the sum of the energetic interactions of the *groups*, rather than the molecules themselves.

First, the pure component parameters, q_i and r_i, are defined as:

$$q_i = \sum_{g \in G} v_{gi} Q_g \quad \text{and} \quad r_i = \sum_{g \in G} v_{gi} R_g \tag{24.38}$$

where G is the set of all groups in the system; v_{gi} is the number of groups, g, contained in component i; Q_g is the group area parameter; and R_g is the group volume parameter.

The residual part of the UNIFAC equation is given by the following expression:

$$\begin{aligned} \ln \gamma_i^R = \sum_{g \in G} Q_g v_{gi} \Bigg\{ & - \sum_{m \in G} \frac{\Psi_{gm} \sum_{j \in N} Q_m v_{mj} x_j}{\sum_{l \in G} \Psi_{lm} \sum_{k \in N} Q_l v_{lk} x_k} \\ & - \ln \left(\sum_{m \in G} \Psi_{mg} \sum_{j \in N} Q_m v_{mj} x_j \right) \\ & + \ln \left(\sum_{j \in N} q_j x_j \right) \\ & + \ln \left(\sum_{m \in G} \Psi_{mg} Q_m v_{mi} \right) \\ & - \ln q_i \\ & + \sum_{m \in G} \frac{\Psi_{gm} Q_m v_{mi}}{\sum_{l \in G} \Psi_{lm} Q_l v_{li}} \Bigg\} \end{aligned} \tag{24.39}$$

where the temperature-dependent group-interaction parameter, Ψ_{gm}, is given by

$$\Psi_{gm} = \exp\left(-\frac{a_{gm}}{T}\right) \tag{24.40}$$

and the group-interaction parameter, a_{gm}, is determined from experimental phase equilibrium data. Similar to the NRTL and UNIQUAC equations, the group interaction parameters are treated as constants with respect to temperature. By introducing a few substitutions, the residual contribution can be simplified to

$$\begin{aligned} \ln \gamma_i^R = & -\sum_{g\in G} \kappa_{gi} \ln\left(\sum_{j\in N} \alpha_{gj} x_j\right) \\ & + \sum_{g\in G} \kappa_{gi} \ln\left(\sum_{j\in N} q_j x_j\right) \\ & - \sum_{g\in G} \frac{\sum_{j\in N} \beta_{gij} x_j}{\sum_{k\in N} \alpha_{gk} x_k} \\ & + \delta_i \end{aligned} \tag{24.41}$$

where the constants $\kappa_{gi}, \alpha_{gj}, \beta_{gij}$, and δ_i are given by

$$\kappa_{gi} = Q_g v_{gi} \tag{24.42}$$

$$\alpha_{gj} = \sum_{m\in G} \kappa_{mj} \Psi_{mg} \tag{24.43}$$

$$\beta_{gij} = \sum_{m\in G} \kappa_{mi} \kappa_{gj} \Psi_{mg} \tag{24.44}$$

$$\delta_i = \sum_{g\in G} \kappa_{gi} \cdot \ln\left(\frac{\sum_{m\in G} \kappa_{mi} \Psi_{mg}}{q_i}\right) \tag{24.45}$$

$$+ \sum_{g\in G} \kappa_{gi} \sum_{m\in G} \frac{\kappa_{mi} \Psi_{gm}}{\sum_{l\in G} \kappa_{li} \Psi_{lm}}. \tag{24.46}$$

Using the simplified expression for the residual contribution, the N-ary formulation using the UNIFAC equation is given by:

$$
\begin{aligned}
\min_{\boldsymbol{x},s} \quad & s \\
\text{s.t.} \quad & (\tfrac{z}{2}q_i - 1)\ln(\sum_{j\in N} r_j x_j) - \tfrac{z}{2}q_i \ln(\sum_{j\in N} q_j x_j) - r_i \frac{\sum_{j\in N} l_j x_j}{\sum_{j\in N} r_j x_j} \\
& - \sum_{g\in G} \kappa_{gi} \ln\left(\sum_{j\in N} \alpha_{gj} x_j\right) + \sum_{g\in G} \kappa_{gi} \ln\left(\sum_{j\in N} q_j x_j\right) - \sum_{g\in G} \frac{\sum_{j\in N} \beta_{gij} x_j}{\sum_{k\in N} \alpha_{gk} x_k} \\
& + \left(\ln r_i + \tfrac{z}{2} q_i \ln(\tfrac{q_i}{r_i}) + l_i + \delta_i\right) + \tfrac{b_i}{T+c_i} + \ln P - a_i - s \;\leq\; 0, \forall\, i \in N \\
& -(\tfrac{z}{2}q_i - 1)\ln(\sum_{j\in N} r_j x_j) + \tfrac{z}{2}q_i \ln(\sum_{j\in N} q_j x_j) + r_i \frac{\sum_{j\in N} l_j x_j}{\sum_{j\in N} r_j x_j} \\
& + \sum_{g\in G} \kappa_{gi} \ln\left(\sum_{j\in N} \alpha_{gj} x_j\right) - \sum_{g\in G} \kappa_{gi} \ln\left(\sum_{j\in N} q_j x_j\right) + \sum_{g\in G} \frac{\sum_{j\in N} \beta_{gij} x_j}{\sum_{k\in N} \alpha_{gk} x_k} \\
& - \left(\ln r_i + \tfrac{z}{2} q_i \ln(\tfrac{q_i}{r_i}) + l_i + \delta_i\right) - \tfrac{b_i}{T+c_i} - \ln P + a_i - s \;\leq\; 0, \forall\, i \in N \\
& \sum_{i\in N} x_i \;=\; 1 \\
& 0 < \boldsymbol{x} < 1 \\
& s \geq 0.
\end{aligned}
\tag{24.47}
$$

Formulation (24.47) contains several nonconvex terms. As is the case for the UNIQUAC equation, these terms fall into two categories: logarithmic terms, and fractional terms. In addition, since the combinatorial part of the UNIFAC equation is identical to the UNIQUAC equation, the nonconvex terms that arise from this part are also identical. These equations are the first two expressions in (24.33), and the first two expressions in (24.34). The convex lower bounding functions for these terms are given by the corresponding UNIQUAC equations. The nonconvex terms from the residual part are the following:

$$
\ln\left(\sum_{j\in N} \alpha_{gj} x_j\right), \quad \text{and} \quad \ln\left(\sum_{j\in N} q_j x_j\right) \tag{24.48}
$$

and

$$
+\frac{\beta_{gij} x_j}{\sum_{k\in N} \alpha_{gk} x_k}, \quad \text{and} \quad -\frac{\beta_{gij} x_j}{\sum_{k\in N} \alpha_{gk} x_k}. \tag{24.49}
$$

Note that the second logarithmic nonconvex term is identical to one of the nonconvex terms from the combinatorial part, for which we have already derived a convex lower bounding function. Convex underestimators for the remaining terms are derived next.

Convex Underestimating Functions for the UNIFAC Equation. The N-ary azeotrope problem using the UNIFAC equation contains several nonconvex terms. Most of the terms are identical to the nonconvex terms from the UNIQUAC equation, as discussed in Section 24.5.3. Convex lower bounding functions for these terms are derived in section 24.4. Of the remaining nonconvex terms, the first one has the same logarithmic form as discussed for the UNIQUAC equation.

$$\ln\left(\sum_{j\in N}\alpha_{gj}x_j\right).$$

This is a concave term, thus the convex lower bounding function is given by the following equation:

$$\ln\left(\sum_{j\in N}\alpha_{gj}x_j\right) \geq \ln m_g^L + \frac{\ln\left\{\frac{m_g^U}{m_g^L}\right\}}{(m_g^U - m_g^L)}\left(\sum_{j\in N}\alpha_{gj}x_j - m_g^L\right)$$

where

$$\begin{aligned} m_g^U &= \max \sum_{j\in N}\alpha_{gj}x_j \\ &\text{s.t.} \sum_{j\in N} x_j = 1 \\ &\quad x_j^L \leq x_j \leq x_j^U \end{aligned}$$

and,

$$\begin{aligned} m_g^L &= \min \sum_{j\in N}\alpha_{gj}x_j \\ &\text{s.t.} \sum_{j\in N} x_j = 1 \\ &\quad x_j^L \leq x_j \leq x_j^U. \end{aligned}$$

The last two nonconvex terms for the UNIFAC equation are fractional terms :

$$+\frac{\beta_{gij}x_j}{\sum_{k\in N}\alpha_{gk}x_k}, \quad \text{and} \quad -\frac{\beta_{gij}x_j}{\sum_{k\in N}\alpha_{gk}x_k}.$$

For the UNIFAC equation, the parameter β_{gij} is always positive. Therefore, for the positive fractional term, the underestimating function is given by:

$$+\frac{\beta_{gij}x_j}{\sum_{k\in N}\alpha_{gk}x_k} \geq \max\left\{\frac{\beta_{gij}x_j^L}{\sum_{k\in N}\alpha_{gk}x_k} + \frac{\beta_{gij}x_j}{m_g^U} - \frac{\beta_{gij}x_j^L}{m_g^U}, \frac{\beta_{gij}x_j^U}{\sum_{k\in N}\alpha_{gk}x_k} + \frac{\beta_{gij}x_j}{m_g^L} - \frac{\beta_{gij}x_j^U}{m_g^L}\right\}.$$

For the negative fractional term, the underestimating function is given by:

$$-\frac{\beta_{gij}x_j}{\sum_{k\in N}\alpha_{gk}x_k} \geq \max\left\{-\frac{\beta_{gij}x_j}{m_g^L}+\frac{\beta_{gij}x_j^L\cdot\sum_{k\in N}\alpha_{gk}x_k}{m_g^L m_g^U}-\frac{\beta_{gij}x_j^L}{m_g^U},\right.$$
$$\left.-\frac{\beta_{gij}x_j}{m_g^U}+\frac{\beta_{gij}x_j^U\sum_{k\in N}\alpha_{gk}x_k}{m_g^L m_g^U}-\frac{\beta_{gij}x_j^U}{m_g^L}\right\}$$

where,

$$\begin{aligned} m_g^U &= \max \sum_{k\in N}\alpha_{gk}x_k \\ &\text{s.t.} \sum_{k\in N} x_k = 1 \\ & x_k^L \leq x_k \leq x_k^U \end{aligned}$$

and,

$$\begin{aligned} m_g^L &= \min \sum_{k\in N}\alpha_{gk}x_k \\ &\text{s.t.} \sum_{k\in N} x_k = 1 \\ & x_k^L \leq x_k \leq x_k^U. \end{aligned}$$

24.5 COMPUTATIONAL STUDIES

The deterministic global optimization algorithm for enclosing all homogeneous azeotropes in a multicomponent mixture follows the procedure described in section 23.4 of Chapter 23 (also see Harding *et al.* (1997)). The approach was implemented in GAMS on a Hewlett Packard 9000/730 workstation. The solver MINOS5.4 is used to perform local minimizations (Murtagh and Saunders, 1983). All times reported are the total CPU time in seconds needed to obtain all azeotropes in the system. A convergence tolerance of 10^{-5} was used for all examples.

In Section 24.5.1, one example problem is selected and solved using both the Wilson equation and the NRTL equation. In Section 24.5.2, the application for the UNIQUAC equation is demonstrated with one example. Finally, in Section 24.5.3, the UNIFAC equation is used to locate all azeotropes in one example. For extensive computational studies of the discussed approach the reader is directed to the work of Harding *et al.* (1997).

24.5.1 Application to the Wilson and NRTL Models

Example 1 : Acetone/Methyl Acetate/Methanol. A system encountered frequently in the literature of phase equilibria problems is the Acetone/Methyl Acetate/Methanol system, (Fidkowski *et al.*, 1993). Experimental data in Table 24.1 show that this system contains three binary azeotropes and

one ternary azeotrope. In this example, the Wilson equation and the NRTL equation are used to model the liquid phase activity coefficients. As can be seen in Table 24.2, the Wilson equation predicted all four azeotropes. The Acetone/Methanol and Methyl Acetate/Methanol binary azeotrope predictions are close to the experimental results. On the other hand, the prediction for the Acetone/Methyl Acetate azeotrope and the ternary azeotrope are significantly different from the experimental results. Note, however, that our results with the Wilson equation are virtually identical to the results of Fidkowski *et al.* (1993). This verifies that the global optimization method will find all homogeneous azeotropes that can be predicted by a given activity coefficient equation and set of parameters. In general, the NRTL equation gave better results for the prediction of the binary azeotropes, but the NRTL equation does not predict a ternary azeotrope.

The computational requirements for calculating the binary azeotropes were under one CPU second for both activity coefficient models. In Table 24.2, the column *NLPs* refers to the number of nonlinear subproblems that were solved in order to converge to the global solution. For the binary systems in this example, about forty nonlinear subproblems were solved per CPU second. In the ternary system, about thirty subproblems were solved per CPU second, due to the larger size of the problem. For the ternary azeotrope with the Wilson model, the cpu requirement was 7.39 CPU seconds.

Experimental Data for Acetone (1) – Methyl Acetate (2) – Methanol (3) $P = 1.0$ atm				
Azeotrope	x_1	x_2	x_3	T (deg C)
Data from Gmehling *et al.* (1994)				
1-2	0.610	0.390	—	56.10
1-3	0.785	—	0.215	55.20
2-3	—	0.680	0.320	54.00
1-2-3 †	0.048	0.620	0.332	55.10

Table 24.1. Experimental Data for Example 1

In the the second part of this example, the k-ary azeotrope problem defined in formulation (24.13) was used with the Wilson activity coefficient equation to locate all binary and ternary azeotropes in the system. The results for this example are listed in Table 24.3.

While this method successfully located all of the azeotropes, it was much slower than the N-ary formulation (278.75 CPU seconds). A total of 272 NLP subproblems were solved in order to find all of the azeotropes using the N-ary

Results for Acetone (1) – Methyl Acetate (2) – Methanol (3) Find all N-ary Azeotropes: $P = 1.0$ atm						
Azeotrope	x_1	x_2	x_3	T (deg C)	CPU (sec)	NLPs
Wilson Equation						
1-2	0.532	0.468	—	55.675	0.47	21
1-3	0.747	—	0.253	54.505	0.37	17
2-3	—	0.677	0.323	54.356	0.43	17
1-2-3	0.272	0.465	0.253	54.254	7.39	217
NRTL Equation						
1-2	0.655	0.345	—	55.557	0.94	29
1-3	0.777	—	0.223	55.117	0.57	19
2-3	—	0.663	0.337	53.545	0.52	19
1-2-3	No ternary azeotrope was found using the NRTL Equation					
Wilson Equation solution from Fidkowski *et al.* (1993)						
AMa	0.5396	0.4606	—	55.66		
AM	0.7480	—	0.2520	54.48		
MaM	—	0.6764	0.3236	54.36		
AMaM	0.2852	0.4538	0.2610	54.25		

Table 24.2. N-ary Solutions for Example 1

formulation. However, the k-ary formulation required 1763 subproblems, and only six subproblems per CPU second could be solved. This is due to the more complicated nonconvex expressions in the k-ary formulation. The convex underestimators in this case are not as tight as for the N-ary azeotrope problem. As a result, it is more efficient to formulate a *series* of N-ary problems to find all azeotropes in a multicomponent mixture, rather than solve a single k-ary problem.

24.5.2 Application to the UNIQUAC Model

Example 2 : Benzene/Ethanol/Water. In this example, the UNIQUAC formulation is used to find all azeotropes in the system of benzene/ethanol/water. This example has been widely examined due to the importance of the azeotropic

Solution for Acetone (1) – Methyl Acetate (2) – Methanol (3) Find all k-ary Azeotropes: Wilson Equation $P = 1.0$ atm CPU = 278.75 sec NLP Subproblems Solved = 1763				
Azeotropes	x_1	x_2	x_3	T (deg C)
1-2	0.532	0.468	—	55.675
1-3	0.747	—	0.253	54.505
2-3	—	0.677	0.323	54.356
1-2-3	0.272	0.465	0.263	54.254

Table 24.3. k-ary Solutions for Example 1

distillation process used to separate ethanol and water using benzene as an entrainer. The N-ary azeotrope method located all three binary azeotropes. The benzene-ethanol and ethanol-water predictions are close to the experimental values. The benzene-water binary azeotrope prediction is not as accurate. It has been reported in the experimental literature that this system contains a ternary heterogeneous azeotrope, but no ternary homogeneous azeotropes. However, the UNIQUAC equation did predict a ternary homogeneous azeotrope. The results are included in Table 24.4. The computational requirements were similar to the Wilson and NRTL examples. Between thirty and forty NLP subproblems per CPU second were solved for the binary examples, and around sixteen NLP subproblems per second for the ternary system. This is slightly slower than Example 1 for the Wilson and NRTL examples.

24.5.3 Application to the UNIFAC Model

Example 3 : Ethanol/Methyl-Cyclopentane/Benzene/Hexane. In this example, a quaternary system is studied, using the UNIFAC equation. The experimental azeotrope data are in Table 24.5. Surprisingly, the UNIFAC equation gives better results, shown in Table 24.6, than the UNIQUAC equation. The UNIFAC equation locates all of the binary azeotropes reported in Gmehling *et al.* (1994), and it does not predict the non-existant ternary and quaternary azeotropes that the UNIQUAC equation did. The CPU times are of the same order of magnitude as the other thermodynamic models.

Summary

This chapter introduced a deterministic global optimization method for computing all N-ary and k-ary ($k \leq N$) homogeneous azeotropes in mixtures containing N components. This method is based on a branch and bound framework to iteratively solve a re-formulation of the original problem in which the nonconvex constraints are replaced by valid convex underestimating functions. Nonconvex terms for the saturated vapor pressure equation $\ln P_i^{sat}$, the Wilson equation, the NRTL equation, the UNIQUAC equation, and the UNIFAC equation were identified and convex underestimating terms were derived for each. The novel contribution of this approach is that it offers a theoretical guarantee of enclosing all homogeneous azeotropes with modest computational effort. The effectiveness of this method was demonstrated for several chemical systems and three different thermodynamic models for the liquid phase activity coefficients. Additional computational studies for homogeneous azeotropes and comparisons with experimental data can be found in Harding *et al.* (1997). Recently, this approach has been extended to address the determination of all heterogeneous and reactive azeotropes (Harding and Floudas, 1999a). Explicit mathematical formulations of the examples considered in this chapter, as well as several additional problems can be found in the recent book by Floudas *et al.* (1999) on test problems for local and global optimization.

Solution for Benzene (1) – Ethanol (2) – Water (3) Find all N-ary Azeotropes: $P = 1.0$ atm						
Azeotrope	x_1	x_2	x_3	T (deg C)	CPU (sec)	NLPs
Experimental Data from Gmehling *et al.* (1994)						
1-2	0.570	0.430	—	64.80		
1-3	0.704	—	0.296	69.30		
2-3	—	0.895	0.105	78.10		
1-2-3	No Homogeneous Azeotropes					
UNIQUAC Equation						
1-2	0.572	0.428	—	67.331	0.94	31
1-3	0.608	—	0.392	61.317	0.84	31
2-3	—	0.886	0.114	78.153	0.84	29
1-2-3	0.604	0.013	0.383	61.583	6.48	107

Table 24.4. Solutions for Example 2 with UNIQUAC

Solution for Ethanol (1) – Methyl-Cyclopentane (2) – Benzene (3) – Hexane (4) Find all N-ary Azeotropes: $P = 1.0$ atm						
Azeotrope	x_1	x_2	x_3	x_4	$T(degC)$	CPU (sec)
UNIQUAC Equation						
1-2	0.316	0.684	—	—	63.489	1.56
1-3	0.441	—	0.559	—	67.292	0.94
1-4	0.334	—	—	0.666	61.265	1.37
2-3	—	0.738	0.262	—	70.367	0.77
2-4	—	0.654	—	0.346	72.892	0.78
3-4	—	—	0.307	0.693	66.919	1.04
1-2-3	0.301	0.545	0.154	—	63.029	3.49
1-2-4	0.344	0.418	—	0.238	64.401	7.94
1-3-4	0.300	—	0.172	0.528	60.669	6.86
2-3-4	—	0.458	0.333	0.209	71.235	5.53
1-2-3-4	0.322	0.322	0.222	0.133	63.558	27.31

Table 24.5. Solutions for Example 3 with UNIQUAC

Solution for Ethanol (1) – Methyl-Cyclopentane (2) – Benzene (3) – Hexane (4) Find all N-ary Azeotropes: $P = 1.0$ atm						
Azeotrope	x_1	x_2	x_3	x_4	T (deg C)	CPU (sec)
UNIFAC Equation						
1-2	0.371	0.629	—	—	60.632	1.86
1-3	0.450	—	0.550	—	67.841	1.16
1-4	0.357	—	—	0.643	59.054	2.04
2-3	—	0.865	0.135	—	71.651	1.60
3-4	—	—	0.065	0.935	68.900	1.48

Table 24.6. Solutions for Example 3 with UNIFAC

References

M. Abate, B.R. Barmish, C. Murillo-Sanchez, and R. Tempo. Application of Some New Tools to Robust Stability Analysis of Spark Ignition Engines : A Case Study. *IEEE Trans. Contr. Syst. Tech.*, 2(1):22, 1994.

D. S. Abrams and J. M. Prausnitz. Statistical Thermodynamics of Liquid Mixtures: A New Expression for the Excess Gibbs Energy of Partly or Completely Miscible Systems. *AIChE J.*, 21(1):116, 1975.

R.A. Abrams and C.T. Wu. Projection and Restriction Methods in Geometric and Related Problems. *J. Opt. Th. Appl.*, 26:59, 1978.

J. Acevedo and E.N. Pistikopoulos. A Parametric MINLP Algorithm for Process Synthesis Problems Under Uncertainty. *I&EC Res.*, 35(1):147–158, 1996.

J. Ackermann, D. Kaesbauer, and R. Muench. Robust Gamma-Stability Analysis in a Plant Parameter Space. *Automatica*, 27:75, 1991.

N. Adhya, M. Tawarmalani, and N. V. Sahinidis. A Lagrangian Approach to the Pooling Problem. *I&EC Res.*, 38:1956–1972, 1999.

C. S. Adjiman. *Global Optimization Techniques for Process Systems Engineering.* PhD thesis, Princeton University, Princeton, N.J., 1998.

C. S. Adjiman, I. P. Androulakis, and C. A. Floudas. *An Implementation of the αBB Global Optimization Algorithm: User's Guide.* Computer–Aided Systems Laboratory, Dept. of Chemical Engineering, Princeton University, NJ, 1997.

C. S. Adjiman, I. P. Androulakis, and C. A. Floudas. Global Optimization of MINLP Problems in Process Synthesis and Design. *Computers and Chemical Engineering Supplement*, 21:S445–S450, 1997c.

C. S. Adjiman, I. P. Androulakis, and C. A. Floudas. A Global Optimization Method, αBB, for General Twice–Differentiable NLPs – II. Implementation and Computational Results. *Comput. chem. engng.*, 22(9):1159–1179, 1998b.

C. S. Adjiman, I. P. Androulakis, and C. A. Floudas. Global Optimization of Mixed-Integer Nonlinear Problems. submitted for publication, 1998d.

C. S. Adjiman, I. P. Androulakis, C. D. Maranas, and C. A. Floudas. A Global Optimisation Method, αBB, for Process Design. *Computers & Chemical Engineering*, 20:S419–S424, 1996.

C. S. Adjiman, S. Dallwig, C. A. Floudas, and A. Neumaier. A Global Optimization Method, αBB, for General Twice–Differentiable NLPs – I. Theoretical Advances. *Comput. chem. engng.*, 22(9):1137–1158, 1998a.

C. S. Adjiman and C. A. Floudas. Rigorous Convex Underestimators for General Twice–Differentiable Problems. *J. of Glob. Opt.*, 9:23–40, 1996.

C. S. Adjiman, C. A. Schweiger, and C. A. Floudas. Mixed-integer nonlinear optimization in process synthesis. In D.-Z. Du and P.M. Pardalos, editors, *Handbook of Combinatorial Optimization.* Kluwer Academic Publishers, 1998c.

A. Aggarwal and C. A. Floudas. Synthesis of general distillation sequences - nonsharp separations. *Comp. & Chem. Eng.*, 14(6):631, 1990.

A. Aggarwal and C. A. Floudas. Synthesis of heat integrated nonsharp distillation sequences. *Comp. & Chem. Eng.*, 16:89, 1992.

E. Aiyoshi and K. Shimizu. Hierarchical Decentralized systems and its new solution by a Barrier Method. *IEEE Trans. Systems, Man and Cybernetics*, SMC-11:444, 1981.

E. Aiyoshi and K. Shimizu. A solution method for the Static Constraint Stackelberg Problem Via Penalty Method. *IEEE Trans. Aut. Control*, AC-29:1111, 1984.

F. A. Al-Khayyal. Jointly Constrained Bilinear Programs and Related Problems: An Overview. *Computers in Mathematical Applications*, 19:53, 1990.

F. A. Al-Khayyal and J. E. Falk. Jointly constrained biconvex programming. *Math. of Oper. Res.*, 8(2):273, 1983.

F. A. Al-Khayyal, R. Horst, and P. M. Pardalos. Global Optimization of Concave Functions Subject to Quadratic Constraints: An Application in Nonlinear Bilevel Programming. *Annals of Oper. Res.*, 34:125, 1992.

R.I. Allgor and P.I. Barton. Mixed Integer Dynamic Optimization. *Computers & Chemical Engineering*, 21:S451–S456, 1997.

N. L. Allinger, Y. H. Yuh, and J-H. Lii. Molecular Mechanics. The MM3 force field for hydrocarbons. *J. Am. Chem. Soc.*, 111:8551–8566, 1989.

G. Anandalingam and D. J. White. A solution method for the Linear Static Stackelberg Problem Using Penalty Method. *IEEE Trans. Aut. Control*, 35:1170, 1990.

T. F. Anderson, D. S. Abrams, and E. A. Grens II. Evaluation of Parameters for Nonlinear Thermodynamic Models. *AIChE J.*, 24(1):20, 1978.

T. F. Anderson and J. M. Prausnitz. Application of the UNIQUAC Equation to Calculation of Multicomponent Phase Equilibria. 1: Vapor-Liquid Equilibria. *Ind. Eng, Chem. Proc. Des. Dev.*, 17:552, 1978.

I. P. Androulakis and C. A. Floudas. Distributed Branch and Bound Algorithms in Global Optimization. In *Parallel Processing of Discrete Problems, (Ed. P. M. Pardalos*, volume 106 of *IMA Volumes in Mathematics and Its Applications, Springer-Verlag*, pages 1–36, 1998.

I. P. Androulakis, C. D. Maranas, and C. A. Floudas. Global minimum potential energy conformations of oligopeptides. *J. Global Optim.*, 11:1–34, 1997a.

I. P. Androulakis, N. N. Nayak, M. G. Ierapetritou, D. S. Monos, and C. A. Floudas. A predictive method for the evaluation of peptide binding in pocket 1 of HLA-DRB1 via global minimization of energy interactions. *Proteins*, 29:87–102, 1997b.

I. P. Androulakis, V. Visweswaran, and C. A. Floudas. Distributed Decomposition-Based Approaches in Global Optimization. In *Proceedings of the State of the Art in Global Optimization: Computational Methods and Applications (Eds. C.A. Floudas and P.M. Pardalos)*, Kluwer Book Series on Nonconvex Optimization and Its Applications, pages 285–302. Kluwer Academic Publishers, 1996.

I.P. Androulakis, C. D. Maranas, and C. A. Floudas. αBB : a global optimization method for general constrained nonconvex problems. *Journal of Global Optimization*, 7:337–363, 1995.

C. B. Anfinsen, E. Haber, M. Sela, and F. H. White Jr. The kinetics of formation of native ribonuclease during oxidation of the reduced polypeptide chain. *Proc. Natl. Acad. Sci. USA*, 47:1309–1314, 1961.

V. Y. Aristovich and E. I. Stepanova. Determination of the Existence and Composition of Multicomponent Azeotropes by Calculation from Data for Binary Systems. *Zh. Prikl. Khim. (Leningrad)*, 43:2192–2200, 1970.

J. D. Augspurger and H. A. Scheraga. An Efficient, Differentiable Hydration Potential for Peptides and Proteins. *J. Comp. Chem*, 17:1549–1558, 1996.

M. Avriel. *Nonlinear Programming : Analysis and Methods.* Prentice-Hall Inc., Englewood Cliffs, New Jersey, 1976.

M. Avriel. *Advances In Geometric Programming.* Plenum Publishing Corporation, New York, NY, 1980.

M. Avriel, R. Dembo, and U. Passy. Solution of Generalized Geometric Programs. *SIAM Intern. J. Numer. Methods Engnr.*, 26:291, 1975.

M. Avriel and D.J. Wilde. Optimal Condenser Design by Geometric Programming. *I & EC Proc. Des. Dev.*, 6:256, 1967.

M. Avriel and A. C. Williams. An extension of geometric programming with applications in engineering optimization. *Journal of Engineering Mathematics*, 5(3):187–194, 1971.

D. J. Bacon and J. Moult. Docking by least-squares fitting of molecular surface patterns. *J. Mol. Biol.*, 225:849–858, 1992.

L.E. Baker, A.C. Pierce, and K.D. Luks. Gibbs energy analysis of phase equlibria. *Soc. Petro. Eng. J.*, page 731, October 1982.

T.E. Baker and L.S. Lasdon. Successive linear programming at Exxon. *Mgmt. Sci.*, 31(3):264, 1985.

P. Ballone and P. Milani. Simulated Annealing of Carbon Clusters. *Phys. Rev.*, 42:3905, 1990.

J. R. Banga and W. D. Seider. Global Optimization of Chemical Processes Using Stochastic Algorithms. In *Proceedings of the State of the Art in Global Optimization: Computational Methods and Applications (Eds. C.A. Floudas and P.M. Pardalos)*, Kluwer Book Series on Nonconvex Optimization and Its Applications. Kluwer Academic Publishers, 1996.

A.P. Barbosa-Povoa and S. Macchietto. Detailed Design of Multipurpose Batch Plants. *Computers & Chemical Engineering*, 18(11/12):1013–1042, 1994.

J.F. Bard. An Algorithm for Solving the General Bilevel Programming Problem. *Math. Oper. Res.*, 8:260, 1983.

J.F. Bard. Optimality Conditions for the Bilevel Programming Problem. *Nav. Res. Log. Q.*, 31:13, 1984.

J.F. Bard. Convex Two-Level Optimization. *Math. Prog.*, 40:15, 1988.

J.F. Bard and J.E. Falk. A separable programming approach to the linear complementarity problem. *Comput. and Oper. Res.*, 9(2):153, 1982.

J.F. Bard and J.T. Moore. A Branch and Bound Algorithm for the Bilevel Programming Problem. *SIAM J.*, 11:281, 1990.

Y. Bard. *Nonlinear Parameter Estimation.* Academic Press, New York, 1974.

Y. Bard and L. Lapidus. Kinetics Analysis by Ditigal Parameter Estimation. *Catalysis Reviews*, 2(1):67, 1968.

B. R. Barmish, C. A. Floudas, H. V. Hollot, and R. Tempo. A Global Linear Programming Solution to Some Open Robustness Problems Including Matrix Polytope Stability. *Proceedings of the ACC 95, Seattle, June 21-23*, 1995.

B.R. Barmish. *New Tools for Robustness of Linear Systems.* MacMillan Publishing Company, New York, NY, 1994.

P.I. Barton, R.J. Allgor, and W.F. Feehery. Dynamic Optimization in a Discontinuous World. *I&EC Res.*, 37(3):966–981, 1998.

M. S. Bazaraa, H. D. Sherali, and C. M. Shetty. *Nonlinear Programming: Theory and Algorithms, 2nd Edition.* J. Wiley, New York, 1993.

E. M. L. Beale. Integer programming. In *The State of the Art in Numerical Analysis*, pages 409–448. Academic Press, 1977.

P.A. Beck and J.G. Ecker. A Modified Concave Simplex Algorithm for Geometric Programming. *J. Opt. Th. Appl.*, 15:189, 1975.

N. Bekiaris, G.A. Meski, C.M. Radu, and M. Morari. Multiple Steady States in Homogeneous Azeotropic Distillation. *Ind. Eng. Chem. Res.*, 23(9):2023, 1993.

O. Ben-Ayed and C.E. Blair. Computational Difficulties of Bilevel Linear Programming. *Oper. Res. Tech. Notes*, page 556, 1989.

A. Ben-Tal, G. Eiger, and V. Gershovitz. Global Minimization by Reducing the Duality Gap. *Mathematical Programming*, 63:193–212, 1994.

J. F. Benders. Partitioning procedures for solving mixed-variables programming problems. *Numer. Math.*, 4:238, 1962.

O. Berman and N. Ashrafi. Optimization models for reliability of modular software systems. *IEEE Transactions on Software Engineering*, 19(11):1119–1123, 1993.

D.L. Bertsekas, G.S. Lower, N.R. Sandell, and T.A. Posbergh. Optimal short term scheduling of large-scale power systems. *IEEE Transactions on Automatic Control*, AC–28:1, 1983.

R. Bhargava and V. Hlavacek. Experience with Adopting One-Parameter Imbedding Methods Toward Calculation of Countercurrent Separation Processes. *Chem. Eng. Commun.* 28:165, 1984.

W.F. Bialas and M.H. Karwan. Two-Level Linear Programming. *Mang. Sci.*, 30:1004, 1984.

D.B. Birewar and I.E. Grossmann. Incorporating Scheduling in the Optimal Design of Multiproduct Batch Plants. *Comput. chem. engng.*, 13:141–161, 1989.

D.B. Birewar and I.E. Grossmann. Simultaneous synthesis, sizing and scheduling of multiproduct batch plants. *Comput. chem. engng.*, 29(11):2242, 1990.

G.E. Blau and D.J. Wilde. Generalized Polynomial Programming. *Can. J. Chem. Eng.*, 47:317, 1969.

G.E. Blau and D.J. Wilde. A Lagrangian Algorithm for Equality Constrained Generalized Polynomial Optimization. *AIChE J.*, 17:235, 1971.

U. Block and B. Hegner. Development and application of a simulation model for three-phase distillation. *AIChE J.*, 22(3):582, 1976.

J.A. Bloom. Solving an electricity generating capacity expansion planning problem by generalized benders' decomposition. *Oper. Res.*, 31(5):84, 1983.

H. J. Böhm. The computer program Ludi: A new method for the de novo design of enzyme inhibitors. *J. Comput. Aided Mol. Design*, 6:61–78, 1992a.

H. J. Böhm. Ludi: Rule-based automatic design of new substituents for enzyme inhibitor leads. *J. Comput. Aided Mol. Design*, 6:593–606, 1992b.

H.U. Bohmer and S.D. Peyerimhoff. Stability and Structure of Singly-Charged Argon Clusters : A Monte-Carlo Simulation. *Z. Phys. D - Atoms, Molecules and Clusters*, 11:239, 1989.

I. M. Bomze, T. Csendes, R. Horst, and P. M. Pardalos. *Developments in Global Optimization.* Nonconvex Optimization and its Applications. Kluwer Academic Publishers, 1997.

B. Borchers and J. E. Mitchell. An Improved Branch and Bound Algorithm for Mixed Integer Nonlinear Programs. Technical Report 200, Renssellaer Polytechnic Institute, 1991.

J. Bradley. *An Algorithm for the Numerical Solution of Prorotype Geometric Programs.* Institute of Industrial Research and Standards, Dublin, Ireland, 1973.

H. I. Britt and R. H. Luecke. The Estimation of Parameters in Nonlinear Implict Models. *Technometrics*, 15(2):233, 1973.

A. Brooke, D. Kendrick, and A. Meeraus. *GAMS – Release 2.25. A User's Guide*, 1992.

B. Brooks, R. Bruccoleri, B. Olafson, D. States, S. Swaminathan, and M. Karplus. CHARMM: A Program for Macromolecular Energy Minimization and Dynamics Calculations. *J. Comp. Chem.*, 4:187–217, 1983a.

L.G. Bullard and L.T. Biegler. Iterative Linear Programming Strategies for Constrained Simulation. *Comput. chem. engng.*, 15(4):239, 1991.

U. Burkert and N.L. Allinger. *Molecular Mechanics.* American Chemical Society, Washington DC, 1982.

S. K. Burley and G. A. Petsko. Aromatic–Aromatic Interaction: A Mechanism of Protein Structure Stabilization. *Science*, 229:23–28, 1985.

G.D. Byrne and B.A. Baird. Distillation Calculations Using a Locally Parameterized Continuation Method. *Comput. chem. engng.*, 9:593, 1985.

R. P. Byrne and I. D. L. Bogle. An accelerated interval method for global optimisation. *Computers and Chemical Engineering Supplement*, 20:S49–S54, 1996a.

R. P. Byrne and I. D. L. Bogle. Solving Nonconvex Process Optimization Problems using Interval Subdivision Algorithms. In I. E. Grossmann, editor, *Global Optimization in Engineering Design*, Kluwer Book Series in Nonconvex Optimization and its Applications, pages 155–174. Kluwer Academic Publishers, 1996b. Chapter 5.

A. Calfisch, A. Miranker, and M. Karplus. Multiple copy simultaneous search and construction of ligands in binding sites: application of inhibitors of HIV-1 aspartic proteinase. *J. Med. Chem.*, 36:2142–2164, 1993.

A. Calfisch, P. Niederer, and M. Anliker. Monte Carlo docking of oligopeptides to proteins. *Proteins*, 13:223–230, 1992.

K.V. Camarda and C.D. Maranas. Optimization in Polymer Design Using Connectivity Indices. *I&EC Res.*, 38:1884–1892, 1999.

A. Camerman, D. Mastropaolo, I. Karle, J. Karle, and N. Camerman. Crystal Structure of Leucine–enkephalin. *Nature*, 306:447–450, 1983.

W. Candler and R. Townsley. A Linear Two-Level Programming Problem. *Comp. Oper. Res.*, 9:59, 1982.

M. F. Cardoso, R. L. Salcedo, S. Feyo de Azevedo, and D. Barbosa. A simulated annealing approach to the solution of MINLP problems. *Computers and Chemical Engineering*, 21(12):1349–1364, 1997.

E. Di Cera. *Thermodynamic Theory of Site-Specific Binding Processes in Biological Macromolecules.* Cambridge Univ. Press, 1995.

H.S. Chan and K.A. Dill. Protein Folding in the Landscape Perspective: Chevron Plots and non-Arrhenius Kinetics. *PROTEINS*, 30:2–33, 1998.

R. G. Chapman and S. P. Goodwin. A General Algorithm for the Calculation of Azeotropes in Fluid Mixtures. *Fluid Phase Equilibria*, 85:55–69, 1993.

C.R. Chavez, J.D. Seader, and T.L. Wayburn. Multiple Steady State Solutions for Interlinked Separation Systems. *Ind. Eng. Chem. Fundam.*, 25:566, 1986.

H.S. Chen and M.A. Stadtherr. On Solving Large Sparse Nonlinear Equation Systems. *Comput. chem. engng.*, 8:1, 1984.

J.J.J. Chen. Comments on improvements on a replacement for the logarithmic mean. *Chem. Eng. Sci.*, 42:2488–2489, 1987.

J. Cherfils, S. Duquerroy, and J. Janin. Protein-protein recognition analyzed by docking simulation. *Proteins*, 11:271–280, 1991.

B. W. Church, M. Orešič, and D. Shalloway. Tracking metastable states to free-energy global minima. In *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, volume 23, pages 41–64. American Mathematical Society, 1996.

N. Churi and L. E. K. Achenie. Novel mathematical programming model for computer aided molecular design. *Industrial & Engineering Chemistry Research*, 35(10):3788–3794, 1996.

N. Churi and L. E. K. Achenie. The Optimal Design of Refrigerant Mixtures for a Two-Evaporator Refrigeration System. *Comput. chem. engng.*, 21(13):349–354, 1997.

A. R. Ciric and C. A. Floudas. A retrofit approach of heat exchanger networks. *Comput. chem. engng.*, 13(6):703, 1989.

A. R. Ciric and C. A. Floudas. A mixed-integer nonlinear programming model for retrofitting heat exchanger networks. *I&EC Res.*, 29:239, 1990.

A.R. Ciric and D.Y. Gu. Synthesis of Nonequilibrium Reactive Distillation Processes by MINLP Optimization. *AIChE J.*, 40(9):1479–1487, 1994.

A.R. Ciric and S.G. Huchette. Multiobjective Optimization Approach to Sensitivity Analysis Waste Treatment Costs in Discrete Process Synthesis and Optimization Problems. *I&EC Res.*, 32(11):2636–2646, 1993.

P.A. Clark and A.W. Westerberg. A Note on the Optimality Conditions for the Bilevel Programming Problem. *Nav. Res. Log.*, 35:413, 1988.

M. L. Connolly. Analytical molecular surface calculation. *J. Appl. Cryst.*, 16:548–558, 1983.

M. L. Connolly. Analytical Molecular Surface Calculations. *J. Appl. Cryst.*, 16:548–558, 1983.

M. L. Connolly. Solvent accessible surfaces of proteins and nucleic acids. *Science*, 221:709–713, 1983.

M. L. Connolly. Shape complementarity at the hemoglobin $\alpha_1\beta_1$ subunit interface. *Biopolymers*, 25:1229–1247, 1986.

J. H. Conway and N. J. A. Sloane. *Sphere Packings, Lattices and Groups.* Springer-Verlag, 1988.

T.H. Cormen, C.E. Leiserson, and R.L. Rivest. *Introduction to Algorithms.* McGraw–Hill Book Company, New York, NY, 1989.

H. Cornelius and R. Lohner. Computing the range of values of real functions with accuracy higher than second order. *Computing*, 33:331–347, 1984.

H.S.M. Coxeter. *Introduction to Geometry.* John Wiley & Sons, New York, NY, 1961.

J. P. Crouzeix, J. E. Martinez-Legaz, and M. Volle. *Generalized Convexity, Generalized Monotonicity : Recent Results.* Nonconvex Optimization and its Applications. Kluwer Academic Publishers, 1998.

T. Csendes and D. Ratz. Subdivision Direction Selection in Interval Methods for Global Optimization. *J. Global Optim.*, 7(2):183, 1995.

M.M. Daichendt and I.E. Grossmann. Preliminary Screening for the MINLP Synthesis of Process Systems : I. Aggregation Techniques. *Comput. chem. engng.*, 18:663, 1994a.

M.M. Daichendt and I.E. Grossmann. Preliminary Screening for the MINLP Synthesis of Process Systems : II. Heat Exchanger Networks. *Comput. chem. engng.*, 18:679, 1994b.

P. Dauber-Osguthorpe, V. A. Roberts, D. J. Osguthorpe, J. Wolff, M. Genest, and A. T. Hagler. Structure and energetics of ligand-binding to proteins - Escherichia-coli dihydrofolate reductase trimethoprim, a drug-receptor system. *Proteins*, 4:31–47, 1988.

D. Davidenko. On a New Method of Numerically Integrating a System of Nonlinear Equations. *Dokl. Akad. Nauk USSR*, 88:601, 1953.

R.R.E. de Gaston and M.G. Safonov. Exact Calculation of the Multi-Loop Stability Margin. *IEEE Trans. Automat. Contr.*, 33:68, 1988.

J. P. Dean and G. A. Dervakos. Design of process-compatible biological agents. *Computers and Chemical Engineering, Suppl. A*, 20:S67–S72, 1996.

A. S. Deif. The interval eigenvalue problem. *Z. angew. Math. Mech*, 71(1):61–64, 1991.

R. S. Dembo. A set of geometric programming test problems and their solutions. *Math. Programming*, 10:192–213, 1976.

R.S. Dembo. Optimal Design of a Membrane Separation Process using Signomial Programming. *Math. Prog.*, 15:12, 1978a.

R.S. Dembo. Current State of the Art of Algorithms and Software for Geometric Programming. *J. Opt. Th. Appl.*, 26:149, 1978b.

R. L. DesJarlais, G. L. Seibel, I. D. Kuntz, P. S. Furth, J. C. Alvarez, P. R. Ortiz de Montellano, D. L. Decamp, L. M. Babe, and C. S. Craik. Structure Based Design of Nonpeptide Inhibitors Specific for The Human Immunodeficiency Virus-1 Protease. *Proc. Natl. Acad. Sci. USA*, 87:6644–6648, 1990.

R. L DesJarlais, R. P. Sheridan, G. L. Seibel, J. S. Dixon, I. D. Kuntz, and R. Venkataraghavan. Using shape complementarity as an initial screen in designing ligands for a receptor binding site of known three-dimensional structure. *J. Med. Chem.*, 31:722–729, 1988.

K.A. Dill and H.S. Chan. From Levinthal to Pathways to Funnels. *Nature Struc. Biol.*, 4:10–19, 1997.

V.D. Dimitriadis and E.N. Pistikopoulos. Flexibility Analysis of Dynamic Systems. *I&EC Res.*, 34(12):4451–4462, 1995.

L.C.W. Dixon and G.P. Szego. *Towards global optimisation.* North-Holland, Amsterdam, 1975.

L.C.W. Dixon and G.P. Szego. *Towards global optimisation 2.* North-Holland, Amsterdam, 1978.

M.C. Dorneigh and N.V. Sahinidis. Global Optimization Algorithms for Chip Layout and Compaction. *Engineering Optimization*, 25(2):131–154, 1995.

V. G. Dovi and O. Paladino. Fitting of Experimental Data to Implicit Models Using a Constrained Variation Method. *Comput. chem. engng.*, 13(6):731, 1989.

D. Z. Du and P. M. Pardalos. *Minimax and Applications.* Nonconvex Optimization and its Applications. Kluwer Academic Publishers, 1995.

V. Dua and E.N. Pistikopoulos. Optimization Techniques for Process Synthesis and Material Design Under Uncertainty. *Chem. Eng. Res. Des.*, 76(A3):408–416, 1998.

I.S. Duff, J. Nocedal, and J.K. Reid. The Use of Linear Programming for the Solution of Sparse Sets of Nonlinear Equations. *SIAM J. Sci. Stat. Comp.*, 8:99, 1987.

R.J. Duffin. Cost Minimization Problems Treated by Geometric Means. *Oper. Res.*, 10:668, 1962a.

R.J. Duffin. Complementary Geometric Programming. *SIAM J. Appl. Math.*, 19:125, 1970.

R.J. Duffin. Linearizing Geometric Problems. *SIAM Review*, 12:211, 1970.

R.J. Duffin and E.L. Peterson. Duality Theory for Geometric Programming. *SIAM J. Appl. Math.*, 14:1307, 1966.

R.J. Duffin and E.L. Peterson. Reverse Geometric Programming Treated by Harmonic Means. *Indiana Univ. Math. J.*, 22:531, 1972.

R.J. Duffin, E.L. Peterson, and C. Zener. *Geometric Programming.* John Wiley & Sons, New York, NY, 1967.

M. A. Duran and I. E. Grossmann. An outer approximation algorithm for a class of mixed-integer nonlinear programs. *Mathematical Programming*, 36:307, 1986a.

A. Duvedi and L.E.K. Achenie. On the design of environmentally benign refrigerant mixtures : A Mathematical Programming Approach. *Computers & Chemical Engineering*, 21(8):915–923, 1997.

J.G. Ecker and R.D. Wiebking. Optimal Design of a Dry Type Natural Draft Cooling Tower by Geometric Programming. *J. Opt. Th. Appl.*, 26:305, 1978.

T. G. W. Epperly, M. G. Ierapetritou, and E. N. Pistikopoulos. On the Global and Efficient Solution of Stochastic Batch Plant Design Problems. *Comput. chem. engng.*, 21:1411–1431, 1997.

T. G. W. Epperly and E. N. Pistikopoulos. A reduced space branch-and-bound algorithm for global optimization. *Journal of Global Optimization*, 11:287–311, 1997.

T. G. W. Epperly and R. Swaney. Branch and Bound for Global NLP : New Bounding LP. In I. E. Grossmann, editor, *Global Optimization in Engineering Design*, Kluwer Book Series in Nonconvex Optimization and its Applications, pages 1–36, 1996a. Chapter 1.

T. G. W. Epperly and R. Swaney. Branch and Bound for Global NLP : Iterative LP Algorithm and Results. In I. E. Grossmann, editor, *Global Optimization in Engineering Design*, Kluwer Book Series in Nonconvex Optimization and its Applications, pages 37–74, 1996b. Chapter 1.

S. Erkoc and S. Katircioglu. Molecular Dynamics Simulation of Gold Microclusters. *Chem. Phys. Lett.*, 147:476, 1988.

W.R. Esposito and C.A. Floudas. Global Optimization in Parameter Estimation of Nonlinear Algebraic Models via the Error-In-Variables Approach. *I&EC Res.*, 35(5):1841–1858, 1998.

W.R. Esposito and C.A. Floudas. Global Optimization in Parameter Estimation of Differential Algebraic Systems. *I&EC Res.*, 1999. submitted for publication.

J. Fabries and H. Renon. Method of Evalutation and Reduction of Vapor-Liquid Equilbrium Data of Binary-Mixtures. *AIChE J.*, 21(4):735, 1975.

J.E. Falk. *Global Solutions of Signomial Programs.* Report, The George Washington Univ., Prog. in Logistics, 1973.

N.M. Faqir and I.A. Karimi. Design of multipurpose batch plants with multiple production routes. In *Proceedings FOCAPD '89, Snowmass, Colorado*, page 451, 1990.

J. Farges, M.F. de Feraudy, B. Raoult, and G. Torchet. Noncrystalline Structure of Argon Clusters. I. Polyicosahedral Structure of Ar_N clusters, $20 < N < 50$. *J. Chem. Phys.*, 78:5067, 1983.

J. Farges, M.F. de Feraudy, B. Raoult, and G. Torchet. Cluster Models made of Double Icosahedron Units. *Surf. Sci.*, 156:370, 1985.

J. Farges, M.F. de Feraudy, B. Raoult, and G. Torchet. Noncrystalline Structure of Argon Clusters. II. Miltilayer Icosahedral Structure of Ar_N clusters, $50 < N < 750$. *J. Chem. Phys.*, 84:3491, 1986.

R. H. Fariss and V. H. Law. An Efficient Computational Technique for the Generalized Application of Maximum Likelihood to Improve Correlation of Experimental Data. *Comput. chem. engng.*, 3:95, 1979.

L. Fejes-Toth. *Regular Figures.* New York: McMillan, 1954.

G.B. Ferraris and E. Tronconi. A Fortran Program for the Solution of Systems of Nonlinear Algebraic Equations. *Comput. chem. engng.*, 10:12, 1986.

G. Fichtner, H.-J. Reinhart, and D.W.T. Rippin. The Design of Flexible Chemical Plants by the Application of Interval Mathematics. *Comput. chem. engng.*, 14:1311–1316, 1990.

Z. T. Fidkowski, M. F. Malone, and M. F. Doherty. Computing Azeotropes in Multicomponent Mixtures. *Computers and Chemical Engineering*, 17(12):1141–1155, 1993.

R. Fletcher, J.A.J. Hall, and W.R. Johns. Flexible retrofit design of multiproduct batch plants. *Comput. chem. engng.*, 15(12):843, 1991.

R. Fletcher and S. Leyffer. Solving mixed integer nonlinear programs by outer approximation. *Mathematical Programming*, 66(3):327–349, 1994.

R. Fletcher and S. Leyffer. Numerical Experience with Lower Bounds for MIQP Branch and Bound. *SIAM J. OPTIM.*, 8(2):604–616, 1998.

C. A. Floudas. *Nonlinear and Mixed Integer Optimization: Fundamentals and Applications.* Oxford University Press, New York, 1995.

C. A. Floudas. Deterministic global optimization in design, control, and computational chemistry. In L.T. Biegler, T.F. Coleman, A.R. Conn, and F.N. Santosa, editors, *IMA Volumes in Mathematics and its Applications : Large Scale Optimization with Applications, Part II*, volume 93, pages 129–184, New York, 1997. Springer-Verlag.

C. A. Floudas. Global Optimization In Design and Control of Chemical Process Systems. *Journal of Process Control*, 1999a.

C. A. Floudas. Recent Advances In Global Optimization for Process Synthesis, Design and Control : Enclosure of All Solutions. *Computers & Chemical Engineering*, 23:S963–S974, 1999b.

C. A. Floudas, A. Aggarwal, and A. R. Ciric. Global optimum search for nonconvex NLP and MINLP problems. *Computers and Chemical Engineering*, 13(10):1117–1132, 1989.

C. A. Floudas and A. R. Ciric. Strategies for overcoming uncertainties in heat exchanger network synthesis. *Comput. chem. engng.*, 13(10):1133, 1989.

C. A. Floudas and I. E. Grossmann. Algorithmic approaches to process synthesis: logic and global optimization. In *FOCAPD'94*, volume 91 of *AICHE Symposium Series*, pages 198–221, 1995.

C. A. Floudas, J.L. Klepeis, and P.M. Pardalos. Global Optimization Approaches In Protein Folding and Peptide Docking. In M. Farach-Colton, F.S. Roberts, M. Vingron, and M. Waterman (Eds.), editors, *DIMACS Series In Discrete Mathematics and Theoretical Computer Science*, volume 47, pages 141–171, 1999b.

C. A. Floudas and P. M. Pardalos. *A Collection of Test Problems for Constrained Global Optimization Algorithms*, volume 455 of *Lecture Notes in Computer Science*. Springer-Verlag, Berlin, Germany, 1990.

C. A. Floudas and P. M. Pardalos. *State of the Art in Global Optimization : Computational Methods and Applications*. Kluwer Academic Publishers, 1996.

C. A. Floudas and P. M. Pardalos. Recent Developments in Global Optimization and their Relevance to Process Design. In *FOCAPD'99*, AICHE Symposium Series, 1999.

C. A. Floudas, P. M. Pardalos, C. S. Adjiman, W. R. Esposito, Z.H. Gümüş, S.T. Harding, J.L. Klepeis, C. Meyer, and C.A. Schweiger. *Handbook of Test Problems in Local and Global Optimization*. Kluwer Academic Publishers, 1999.

C. A. Floudas and V. Visweswaran. A global optimization algorithm (GOP) for certain classes of nonconvex NLPs : I. Theory. *Computers and Chemical Engineering*, 14:1397–1417, 1990.

C. A. Floudas and V. Visweswaran. A primal-relaxed dual global optimization approach. *Journal of Optimization Theory and Applications*, 78(2):187–225, 1993.

C.A. Floudas and A. Aggarwal. A decomposition strategy for global optimum search in the pooling problem. *OSRA Journal on Computing*, 2(3), 1990.

C.A. Floudas and P.M. Pardalos. *Recent advances in global optimization*. Princeton Series in Computer Science. Princeton University Press, Princeton, New Jersey, 1992.

J. Fortuny-Amat and B. McCarl. A representation and Economic Interpretation of a Two-Level Programming problem. *J. Oper. Res. Soc.*, 32:783, 1981.

Aa. Fredenslund, R.L. Jones, and J.M. Prausnitz. Group Contribution Estimation of Activity Coefficients in Non-ideal Mixtures. *AIChE J.*, 21:1086, 1975.

D.L. Freeman and J.D. Doll. Quantum Monte Carlo Study of the Thermodynamic Properties of Argon Clusters. *J. Chem. Phys.*, 82:462, 1984.

D. H. Fremont, M. Matsumura, E. A. Stura, P. A. Peterson, and I. A. Wilson. Crystal structures of two viral peptides in complex with murine MHC class I H-2 K^b. *Science*, 257:919–927, 1992.

X. Fu, C. Bono, S. Woulfe, C. Swearingen, N. Summers, F. Sinigaglia, A. Sette, B. Schwartz, and R.W. Carr. Pocket 4 of the HLA-DR Molecule is a Major Determinant of T cell Recognition of Peptide. *Journal of Experimental Medicine*, 181:915–926, 1995.

B. Ganish, D. Horsky, and K. Srikanth. An approach to optimal positioning of a new product. *Management Sci.*, 29:1277, 1983.

C.B. Garcia and W.I. Zangwill. *Pathways to Solutions, Fixed Points and Equilibria.* Prentice–Hall, 1981.

M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* Freeman, San Francisco, CA, 1979.

I.L. Garzon, X.P. Long, R. Kawai, and J.H. Weare. Structure and Dynamics of Lennard-Jones Clusters with Impurities. *Z. Phys. D - Atoms, Molecules and Clusters*, 12:81, 1989b.

A. M. Geoffrion. Generalized Benders decomposition. *Journal of Optimization Theory and Applications*, 10:237–260, 1972.

M. C. Georgiadis, G. E. Rotstein, and S. Macchietto. Optimal layout design in multipurpose batch plants. *Industrial & Engineering Chemistry Research*, 36(11):4852–4863, 1997.

J.C. Geromel and M.R. Belloni. Nonlinear programs with complicating variables: theoretical analysis and numerical experience. *IEEE Trans. Syst. Man. Cynernetics*, SMC–16:231, 1986.

S. Gerschgorin. Über die Abgrenzung der Eigenwerte einer Matrix. *Izv. Akad. Nauk SSSR, Ser. fiz.-mat.*, 6:749–754, 1931.

P. E. Gill, W. Murray, M. A. Saunders, and M. H. Wright. *NPSOL 4.0 User's Guide.* Systems Optimization Laboratory, Dept. of Operations Research, Stanford University, CA., 1986.

L. Glasser and H. A. Scheraga. Calculations on Crystal Packing of a Flexible Molecule, Leu-Enkephalin. *J. Mol. Biol.*, 199:513–524, 1988.

J. Gmehling, J. Menke, J. Krafczyk, and K. Fischer. *Azeotropic Data.* VCH, 1994.

J. Gmehling, D. Tiegs, and U. Knipp. A Comparison of the Predictive Capability of Different Group Contribution Methods. *Fluid Phase Equilibria*, 54:147–165, 1990.

P. J. Goodford. A computational procedure for determining energetically favorable binding sites on biologically important molecules. *J. Med. Chem.*, 28:849–857, 1985.

D. S. Goodsell and A. J. Olson. Automated docking of substrates to proteins by simulated annealing. *Proteins*, 8:195–202, 1990.

W. H. Graham, E. S. Carter II, and R. P. Hicks. Conformational Analysis of Met–Enkephalin in Both Aqueous Solution and in the Presence of Sodium Dodecyl Sulfate Micelles Using Multidimensional NMR and Molecular Modeling. *Biopolymers*, 32:1755–1764, 1992.

R. E. Griffith and R. A. Stewart. A nonlinear programming technique for the optimization of continuous processesing systems. *Manag. Sci.*, 7:379, 1961.

P. D. J. Grootenhuis and P. A. Kollman. Crown ether-neutral molecule interactions studied by molecular mechanics and free energy perturbation calculations. Near quantitative agreement between theory and experimental binding free energies. *J. Am. Chem. Soc.*, 111:4046–4051, 1989.

I. E. Grossmann. *Global Optimization in Engineering Design.* Nonconvex Optimization and Its Applications. Kluwer Academic Publishers, Dordrecht, 1996.

I. E. Grossmann and C. A. Floudas. Active constraint strategy for flexibility analysis in chemical processes. *Comput. chem. engng.*, 11(6):675, 1987.

I. E. Grossmann and R.W.H. Sargent. Optimal Design of Multipurpose Chemical Plants. *Ind. Eng. Chem. Proc. Des. Dev.*, 18:343, 1979.

I.E. Grossmann. MINLP optimization strategies and algorithms for process synthesis. In *Proc. 3rd. Int. Conf. on Foundations of Computer-Aided Process Design*, page 105, 1990.

I.E. Grossmann. Mixed Integer Optimization Techniques for Algorithmic Process Synthesis. In J.L. Anderson, editor, *Advances In Chemical Engineering, Process Synthesis*, volume 23, pages 171–246. Academic Press, 1996b.

I.E. Grossmann, I. Quesada, R. Ramon, and V.T. Voudouris. Mixed-Integer Optimization Techniques for the Design and Scheduling of Batch Processes. In *Proc. of NATO Advanced Study Institute on Batch Process Systems Engineering*, 1992.

K. Gulukota, S. Vajda, and C. DeLisi. Peptide docking using dynamic programming. *J. Comput. Chem.*, 17:418–428, 1996.

A. Gupta and V. Manousiouthakis. Minimum Utility Cost of Mass Exchange Networks with Variable Single Component Supplies and Targets. *I&EC Res.*, 32(9):1937–1950, 1993.

O. K. Gupta and R. Ravindran. Branch and bound experiments in convex nonlinear integer programming. *Management Science*, 31(12):1533–1546, 1985.

O.K. Gupta. *Branch and bound experiments in nonlinear integer programming.* PhD thesis, Purdue University, 1980.

P.C. Haarhoff and J.D. Buys. A New Method for the Optimization of a Nonlinear Function Subject to Nonlinear Constraints. *Comp. J.*, 13:178, 1970.

J. Hammer, C. Bolin, D. Papadopoulos, J. Walsky, J. Higelin, W. Danho, F. Sinigaglia, and Z.A. Nagy. High-affinity binding of short peptides to major histocompatibility complex class II molecules by anchor combinations. *Proc. Natl. Acad. Sci.*, 91:4456, 1994.

J. Hammer, E. Bono, F. Gallazi, C. Belunis, Z. Nagy, and F. Sinigaglia. Precise Prediction of Major Histocompatibility Complex Class II-Peptide Interaction Based on Peptide Side Chain Scanning. *Journal of Experimental Medicine*, 180:2353–2358, 1994.

J. Hammer, F. Gallazzi, E. Bono, R. Karr, J. Guenot, P. Valsasnini, and Z. Nagy. Peptide Binding Specificity of HLA-DR4 Molecules: Correlation

with Rheumatoid Arthritis Association. *Journal of Experimental Medicine*, 181:1847–1855, 1995.

E. Hansen. *Global Optimization using Interval Analysis.* Pure and Applied Mathematics. Marcel Dekker, Inc., 1992.

P. Hansen and B. Jaumard. Reduction of Indefinite Quadratic Programs to Bilinear Programs. *J. Global Optim.*, 2:41, 1992.

P. Hansen, B. Jaumard, and S. Lu. Global Optimization of Univariate Lipschitz Functions : II. New Algorithms and Computational Comparison. *Math. Prog.*, 55:273, 1992b.

P. Hansen, B. Jaumard, and S.H. Lu. Some Further Results on Monotonicity in Globally Optimal Design. *J. Mech. Trans. Autom. Des.*, 111:345, 1989.

P. Hansen, B. Jaumard, and G. Savard. New Branch-and-Bound Rules for Linear Bilevel Programming. *SIAM J. Sci. Stat. Comp.*, 13:1194, 1992.

U. H. E. Hansmann and Y. Okamoto. Prediction of peptide conformation by multicanonical algorithm : New approach to the multiple-minima problem. *J. Comput. Chem.*, 14:1333–1338, 1993.

S. T. Harding and C. A. Floudas. Global optimization in multiproduct and multipurpose batch design under uncertainty. *Industrial & Engineering Chemistry Research*, 36(5):1644–1664, 1997.

S. T. Harding and C. A. Floudas. Phase Stability for Equations of State : A Global Optimization Approach. 1999. submitted for publication.

S. T. Harding and C. A. Floudas. Locating All Heterogeneous and Reactive Azeotropes in Multicomponent Mixtures. 1999a. submitted for publication.

S. T. Harding, C.D. Maranas, C.M. McDonald, and C. A. Floudas. Locating All Homogeneous Azeotropes in Multicomponent Mixtures. *Industrial & Engineering Chemistry Research*, 36(1):160–178, 1997.

I. Harjunkoski. *Application of MINLP methods on a scheduling problem in the paper converting industry.* PhD thesis, Abo Akademi University, Abo, Finland, 1997.

T. N. Hart and R. J. Read. A multiple-start Monte Carlo docking method. *Proteins*, 13:206–222, 1992.

T. N. Hart and R. J. Read. Multiple-start Monte Carlo docking of flexible ligands. In *The Protein Folding Problem and Tertiary Structure Prediction*, pages 71–108. Birkhäuser, 1994.

V. Hatzimanikatis, C. A. Floudas, and J. E. Bailey. Analysis and design of metabolic reaction networks via mixed-integer linear optimization. *AIChE Journal*, 42(5):1277–1292, 1996a.

V. Hatzimanikatis, C. A. Floudas, and J. E. Bailey. Optimization of regulatory architectures in metabolic reaction networks. *Biotechnology and Bioengineering*, 52:485–500, 1996b.

A. Haurie, G. Savard, and D.J. White. A Note on: An Efficient Point Algorithm For a Linear Two-Stage Optimization Problem. *Oper. Res. Tech. Notes*, page 553, 1990.

C.A. Haverly. Studies of the behaviour of recursion for the pooling problem. *ACM SIGMAP Bulletin*, 25:19, 1978.

R. D. Head, M. L. Smyte, T. I. Oprea, C. L. Waller, S. M. Green, and G. R. Marshall. VALIDATE : a new method for receptor-based prediction of binding affinities of novel ligands. *J. Am. Chem. Soc.*, 118:3959–3969, 1996.

R.A. Heidemann and J.M. Mandhane. Some properties of the NRTL equation in correcting liquid-liquid equilibrium data. *Chem. Eng. Sci.*, 28:1213, 1973.

L.J. Hellinckx and M.J. Rijckaert. Minimization of Capital Investment for Batch Processes. *I&EC Proc. Des. Dev.*, 10:422, 1971.

L.J. Hellinckx and M.J. Rijckaert. Optimal Capacities of Production Facilities : An Application of Geometric Programming. *Can. J. Chem. Eng.*, 50:148, 1972.

D. Hertz. The extreme eigenvalues and stability of real symmetric interval matrices. *IEEE Transactions on Automatic Control*, 37(4):532–535, 1992.

V. Hlavacek. Simulation of Countercurrent Separation Processes via Global Approach. *Comput. chem. engng.*, 9:343, 1985.

H.H. Hoang. Topological optimization of networks: A nonlinear mixed integer model employing generalized Benders decomposition. *IEEE Transactions on Automatic Control*, AC–27:164, 1982.

M.R. Hoare. Structure and dynamics of simple microclusters. *Adv. Chem. Phys.*, 40:49, 1979.

M.R. Hoare and J. McInnes. Morphology and Statistical Statics of Simple Microclusters. *Adv. Phys.*, 32:791, 1983.

M.R. Hoare and P. Pal. Physical Cluster Mechanics : Statics and Energy Surfaces for Monoatomic Systems. *Adv. Phys.*, 20:161, 1971b.

D. Hohl, R.O. Jones, R. Car, and M. Parrinelo. Structure of Sulfur Clusters Using Simulated Annealing. *J. Chem. Phys.*, 89:6823, 1988.

K. Holmberg. On the convergence of the cross decomposition. *Mathematical Programming*, 47:269, 1990.

J.D. Honeycutt and H.C. Andersen. Molecular Dynamics Study of Melting and Freezing of Small Lennard-Jones Clusters. *J. Phys. Chem.*, 91:4950, 1987.

A.J. Hopfinger. *Conformational Properties of Macromolecules.* Academic Press, New York, NY, 1973.

R. Horst and P. M. Pardalos. *Handbook of Global Optimization.* Nonconvex Optimization and its Applications. Kluwer Academic Publishers, 1994.

R. Horst, P. M. Pardalos, and N. V. Thoai. *Introduction to Global Optimization.* Nonconvex Optimization and its Applications. Kluwer Academic Publishers, 1995.

R. Horst and H. Tuy. *Global Optimization: Deterministic Approaches.* Springer-Verlag, Berlin, Germany, 1990. 1st Edition.

R. Horst and H. Tuy. *Global Optimization: Deterministic Approaches.* Springer-Verlag, Berlin, 1993. 2nd Edition.

R. Horst and H. Tuy. *Global Optimization: Deterministic Approaches.* Springer-Verlag, Berlin, Germany, 1996. 3rd Edition.

Z. Hua, J. F. Brennecke, and M. A. Stadtherr. Reliable Prediction of Phase Stability Using an Interval-Newton Method. *Fluid Phase Equilibria*, 116:52–59, 1996.

Z. Hua, J. F. Brennecke, and M. A. Stadtherr. Enhanced Interval Analysis for Phase Stability : Cubic Equation of State Models. *Ind. Eng. Chem. Res.*, 37:1519–1527, 1998.

M. G. Ierapetritou and E. N. Pistikopoulos. Batch plant design and operations under uncertainty. *Industrial & Engineering Chemistry Research*, 35(2):772–787, 1996.

M.G. Ierapetritou and E.N. Pistikopoulos. Simultaneous Incorporation of Flexibility and Economic Risk in Operational Planning Under Uncertainty. *Computers & Chemical Engineering*, 18(3):163–189, 1994.

M.G. Ierapetritou and E.N. Pistikopoulos. Design of Multiproduct Batch Plants With Uncertain Demands. *Comput. chem. engng.*, 19, suppl.:S627–S632, 1995.

M.G. Ierapetritou, E.N. Pistikopoulos, and C.A. Floudas. Operational Planning Under Uncertainty. *Computers & Chemical Engineering*, 20(12):1499–1516, 1996.

R. R. Iyer and I. E. Grossmann. Global Optimization of Heat Exchanger Networks with Fixed Configuration for Multiperiod Design. In I. E. Grossmann, editor, *Global Optimization in Engineering Design*, Kluwer Book Series in Nonconvex Optimization and its Applications, pages 289–308, 1996. Chapter 9.

A. N. Jain and M. A. Murcko. Computational methods to predict binding free energy in ligand-receptor complexes. *J. Med. Chem.*, 38:4953–4967, 1995.

J. Janin and S. J. Wodak. Reaction pathway for the quartenary structure change in hemoglobin. *Biopolymers*, 24:509–526, 1985.

T.S. Jardesky, J.C. Gorga, R. Bush, J. Rothbard, J.L. Strominger, and D.C. Wiley. Peptide binding to HLA-DR1: a peptide with most residues substituted to alanine retains MHC binding. *EMBO Jl*, 9:1797, 1990.

T.R. Jefferson and C.H. Scott. Generalized Geometric Programming Applied to Problems of Optimal Control. *J. Opt. Th. Appl.*, 26:117, 1978.

J. Jellinek, T.L. Beck, and R.S. Berry. Solid-Liquid Phase Changes In Simulated Isoenergetic Ar_{13}. *J. Chem. Phys.*, 84:2783, 1986.

F. Jiang and S-H. Kim. "Soft docking" : matching of molecular surface cubes. *J. Mol. Biol.*, 219:79–102, 1991.

W.R. Johns, G. Marketos, and D.W.T. Rippin. The Optimal Design of Chemical Plant to Meet Time-Varying Demands in the Presence of Technical and Commercial Uncertainty. *Trans. Inst. Chem. Eng.*, 56:249–257, 1978.

G. Jones, P. Willett, and R. C. Glen. Molecular recognition of receptor sites using a genetic algorithm with a description of desolvation. *J. Mol. Biol.*, 245:43–53, 1995.

W. L. Jorgensen and J. Tirado-Rives. The OPLS potential functions for Proteins. Energy minimizations for crystals of cyclic peptides and crambin. *J. Am. Chem. Soc.*, 110:1657–1666, 1988.

J.J. Judice and A.M. Faustino. A sequential LCP method for Bilevel Linear Programming. *Annals Oper. Res.*, 34:89, 1992.

R. S. Judson, Y. T. Tan, E. Mori, C. Melius, E. P. Jaeger, A. M Treasurywala, and A. Mathiowetz. Docking flexible molecules: a case study of three proteins. *J. Comput. Chem.*, 16:1405–1419, 1995.

B. Kalitventzeff and F. Marechal. The management of a utility network. In *Process Systems Engineering. PSE '88, Sydney, Australia*, page 223, 1988.

R. B Kearfott. *Rigorous Global Search : Continuous Problems.* Nonconvex Optimization and its Applications. Kluwer Academic Publishers, 1996.

R.B. Kearfott, M. Dawande, K. Du, and Ch. Hu. INTLIB, a Portable FORTRAN 77 Interval Standard Function Library. *ACM Trans. Math. Soft.*, 20(4):447, 1994.

R.B. Kearfott and M. Novoa. Algorithm 681 INTBIS : A Portable Interval Newton Bisection Package. *ACM Trans. Math. Soft.*, 16:152, 1990.

S. K. Kearsley, D. J. Underwood, R. P. Sheridan, and M. D. Miller. Flexibases: a way to enhance the use of molecular docking methods. *J. Comput. Aided Mol. Design*, 8:565–582, 1994.

J. E. Kelley. The cutting plane method for solving convex programs. *Journal of the SIAM*, 8(4):703–712, 1960.

I. Kim, M. J. Liebman, and T. F. Edgar. Robust error-in-variables estimation using nonlinear programming techniques. *AIChE Journal*, 36(7):985, 1990.

P. S. Kim and R. L. Baldwin. Intermediates in the folding reactions of small proteins. *Annu. Rev. Biochem.*, 59:631–660, 1990.

S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by Simulated Annealing. *Science*, 220:671, 1983.

J. L. Klepeis, X. V. Nguyen, and C. A. Floudas. *GLO-FOLD: A package for global optimization using αBB in protein folding.* Princeton University, Princeton, NJ, 1997.

J.L. Klepeis, I.P. Androulakis, M.G. Ierapetritou, and C.A. Floudas. Predicting Solvated Peptide Conformations via Global Minimization of Energetic Atom to Atom Interactions. *Computers & Chemical Engineering*, 22(6):765–788, 1998.

J.L. Klepeis and C.A. Floudas. A Comparative Study of Global Minimum Energy Conformations of Hydrated Peptides. *Journal of Computational Chemistry*, 20(6):636–654, 1999a.

J.L. Klepeis and C.A. Floudas. Free Energy Calculations for Peptides via Deterministic Global Optimization. *Journal of Chemical Physics*, 110(15):7491–7512, 1999b.

J.L. Klepeis, C.A. Floudas, D. Morikis, and J.D. Lambris. Predicting Peptide Structures using NMR Data and Deterministic Global Optimization. *Journal of Computational Chemistry*, 20(16), 1999.

R.W. Klopfenstein. Zeroes of Nonlinear Functions. *J. Assoc. Comput. Mach.*, 8:366, 1961.

O. Knüppel. *PROFIL – Programmer's Runtime Optimized Fast Interval Library.* Technische Informatik III, Technische Universität Hamburg–Harburg, 1993.

A. Kochenberger, R.E.D. Woolsey, and B.A. McCarl. On the Solution of Geometric Programming via Separable Programming. *Oper. Res. Quart.*, 24:285, 1973.

G. R. Kocis and I. E. Grossmann. Relaxation strategy for the structural optimization of process flow sheets. *I&EC Res.*, 26(9):1869, 1987.

G. R. Kocis and I. E. Grossmann. Global optimization of nonconvex MINLP problems in process synthesis. *I&EC Res.*, 27(8):1407, 1988.

G. R. Kocis and I. E. Grossmann. A modelling and decomposition strategy for the MINLP optimization of process flowsheets. *Computers and Chemical Engineering*, 13(7):797–819, 1989.

G. R. Kocis and I. E. Grossmann. Computational experience with DICOPT solving MINLP problems in process systems engineering. *Comput. chem. engng.*, 13:307, 1989a.

A. C. Kokossis and C. A. Floudas. Optimization of complex reactor networks-I. isothermal operation. *Chem. Eng. Sci.*, 45(3):595, 1990.

A. C. Kokossis and C. A. Floudas. Optimal synthesis of isothermal reactor-separator-recycle systems. *Chem. Eng. Sci.*, 46:1361, 1991.

A. C. Kokossis and C. A. Floudas. Optimization of complex reactor networks - II. Nonisothermal Operation. *Chem. Eng. Sci.*, 49(7):1037, 1994a.

A. C. Kokossis and C. A. Floudas. Stability In Optimal Design : Synthesis of Complex Reactor Networks. *aiche*, 40(5):849–861, 1994a.

H. Konno, P. T. Thach, and H. Tuy. *Optimization on Low Rank Nonconvex Structures.* Nonconvex Optimization and its Applications. Kluwer Academic Publishers, 1997.

J. Kostrowicki and H. A. Scheraga. Application of the diffusion equation method for global optimization to oligopeptides. *J. Phys. Chem.*, 96:7442–7449, 1992.

Z. Kravanja and I. E. Grossmann. PROSYN - An MINLP process synthesizer. *Comput. chem. engng.*, 14:1363, 1990.

Z. Kravanja and I.E. Grossmann. A Computational Approach for the Modeling/Decomposition Strategy in the MINLP Optimization of Process Flowsheets with Implicit Models. *I&EC Res.*, 35(6):2065–2070, 1996.

M. Kubicek, H. Hofmann, V. Hlavacek, and J. Sinkule. Multiplicity and Stability in a Sequence of Two Nonadiabatic Nonisothermal CSTRs. *Chem. Engng Sci.*, 35:987, 1980.

J.L. Kuester and J.H. Mize. *Optimization Techniques with Fortran.* McGraw–Hill Book Company, New York, NY, 1973.

I. D. Kuntz, J. M. Blaney, S. J. Oatley, R. Landgridge, and T. E. Ferrin. A geometric approach to macromolecule-ligand interactions. *J. Mol. Biol.*, 161:269–288, 1982.

E. Lahaye. *C.R. Acad. Sci. Paris*, 198:1840, 1934.

M. H. Lambert and H. A. Scheraga. Pattern Recognition in the Prediction of Protein Structure. I. Tripeptide Conformational Probabilities Calculated from the Amino Acid Sequence. *J. Comp. Chem*, 10:770–797, 1989.

L.S. Lasdon, A.D. Waren, S. Sarkar, and F. Palacios-Gomez. Solving the Pooling Problem Using Generalized Reduced Gradient and Successive Linear Programming Algorithms. *ACM SIGMAP Bulletin*, 27:9, 1979.

M. C. Lawrence and P. C. Davis. CLIX: A search algorithm for finding novel ligands capable of binding proteins of known three-dimensional structure. *Proteins*, 12:31–41, 1992.

A. R. Leach. A survey of methods for searching the conformational space of small and medium-sized molecules. In *Reviews in Computational Chemistry*, volume 2, pages 1–55. VCH Publishers, 1991.

A. R. Leach and I. D. Kuntz. Conformational analysis of flexible ligands in macromolecular receptor sites. *J. Comput. Chem.*, 13:733–748, 1992.

B. Lee and F. M. Richards. The interpretation of protein structures : estimation of static accessibility. *J. Mol. Biol.*, 55:379–400, 1971.

R. H. Lee and G. D. Rose. Molecular recognition. I. Automatic identification of topographic surface features. *Biopolymers*, 24:1613–1627, 1985.

P. Leopold, M. Montal, and J. Onuchic. Protein folding funnels : A kinetic approach to the sequence-structure relationship. *Proc. Natl. Acad. Sci. USA*, 89:8721–8725, 1992.

J. Leray and J. Schauder. Topologie et equations fonctionnelles. *Ann. Sci. Ecole Norm Sup.*, 51:45, 1934.

C. Levinthal. Are there pathways to protein folding ? *J. Chem. Phys.*, 65:44–45, 1968.

M. Levitt. Protein Conformation Dynamics and Folding by Computer Simulation. *Ann. Rev. Biophys. Bioeng.*, 11:251, 1982.

M. Levitt. Protein folding by restrained energy minimization and molecular dynamics. *J. Mol. Biol.*, 170:723–764, 1983.

Z. Li and H. A. Scheraga. Monte Carlo minimization approach to the multiple-minima problem in protein folding. *Proc. Natl. Acad. Sci. USA*, 84:6611–6615, 1987.

Z. Li and H. A. Scheraga. Structure and free energy of complex thermodynamic systems. *J. Mol. Struct. (Theochem.)*, 179:333–352, 1988.

W.J. Lin. *Application of continuation and modeling methods to phase equilibrium, steady-state and dynamic process calculations.* PhD thesis, University of Utah, 1988.

M. L. Liu, N.V. Sahinidis, and J. P. Shectman. Planning of Chemical Process Networks via Global Concave Minimization. In I. E. Grossmann, editor, *Global Optimization in Engineering Design*, Kluwer Book Series in Nonconvex Optimization and its Applications, pages 195–230, 1996. Chapter 7.

M.L. Liu and N.V. Sahinidis. Process Planning in a Fuzzy Environment. *Eur. J. Oper. Res.*, 100(1):142–169, 1997.

W.B. Liu and C. A. Floudas. A remark on the GOP algorithm for global optimization. *Journal of Global Optimization*, 3(3):519–521, 1993.

W.B. Liu and C. A. Floudas. Convergence of the GOP algorithm for a large class of smooth optimization problems. *Journal of Global Optimization*, 6:607–611, 1995.

W.B. Liu and C. A. Floudas. Generalized primal-relaxed dual approach for global optimization. *Journal of Optimization Theory and Applications*, 90(2):417–434, 1996.

A. Lucia and J. Xu. Nonconvexity and Descent In Nonlinear Programming. In *Proceedings of the State of the Art in Global Optimization: Computational Methods and Applications (Eds. C.A. Floudas and P.M. Pardalos)*, Kluwer Book Series on Nonconvex Optimization and Its Applications. Kluwer Academic Publishers, 1996.

D. L. Luenberger. *Linear and Nonlinear Programming.* Addison-Wesley, Reading, MA, 2nd edition, 1984.

M.L. Luyben and C.A. Floudas. Analyzing the interaction of design and control, Part 1: A multiobjective framework and application to binary distillation synthesis. *Comput. chem. engng.*, 18(10):933–969, 1994a.

M.L. Luyben and C.A. Floudas. Analyzing the interaction of design and control, Part 2: Reactor–separator–recycle system. *Comput. chem. engng.*, 18(10):971–994, 1994b.

R. W. Maier, J. F. Brennecke, and M. A. Stadtherr. Reliable Computation of Homogeneous Azeotropes. *AIChE J.*, 44:1745–1755, 1998.

V. Manousiouthakis and D. Sourlas. A Global Optimization Approach to Rationally Constrained Rational Programming. *Chem. Eng. Comm.*, 115:127, 1992.

C. D. Maranas. Optimal computer-aided molecular design: a polymer design case study. *Industrial & Engineering Chemistry Research*, 35(10):3403–3414, 1996.

C. D. Maranas. Optimal Molecular Design Under Property Prediction Uncertainty. *AIChE J.*, 43(5):1250–1264, 1997.

C. D. Maranas, I. P. Androulakis, and C. A. Floudas. A Deterministic Global Optimization Approach for the Protein Folding Problem. In *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, volume 23, pages 133–150. American Mathematical Society, 1996.

C. D. Maranas and C.A. Floudas. A global optimization approach for Lennard-Jones microclusters. *Journal of Chemical Physics*, 97(10):7667–7677, 1992.

C. D. Maranas and C.A. Floudas. Global optimization for molecular conformation problems. *Annals of Operations Research*, 42:85–117, 1993.

C. D. Maranas and C.A. Floudas. A deterministic global optimization approach for molecular structure determination. *Journal of Chemical Physics*, 100(2):1247–1261, 1994a.

C. D. Maranas and C.A. Floudas. Global minimum potential energy conformations of small molecules. *Journal of Global Optimization*, 4:135–170, 1994b.

C. D. Maranas and C.A. Floudas. Finding all solutions of nonlinearly constrained systems of equations. *Journal of Global Optimization*, 7(2):143–182, 1995.

C. D. Maranas and C.A. Floudas. Global optimization in generalized geometric programming. *Computers and Chemical Engineering*, 21(4):351–370, 1997.

G. Marketos. *The Optimal Design of Chemical Plant Considering Uncertainty and Changing Circumstances.* PhD thesis, ETH Zurich, 1975.

D. W. Marquardt. An Algorithm for Least-Squares Estimation of Nonlinear Parameters. *J. Soc. Ind. Appl. Math.*, 11(2):431, 1963.

H. Mawengkang and B.A. Murtagh. Solving nonlinear integer programs with large scale optimization software. *Annals of Operations Research*, 5:425, 1986.

G. P. McCormick. Computability of global solutions to factorable nonconvex programs: Part I – Convex underestimating problems. *Math. Programming*, 10:147–175, 1976.

G. P. McCormick. *Nonlinear Programming: Theory, Algorithms and Applications.* J. Wiley, New York, New York, 1983.

G.P. McCormick. Attempts to calculate global solutions of problems that may have local minima. In F.A.Lootsma, editor, *Numerical Methods for Nonlinear Optimization*, page 209. Acad. Press, 1972.

C. M. McDonald and C. A. Floudas. Decomposition based and branch and bound global optimization approaches for the phase equilibrium problem. *Journal of Global Optimization*, 5:205–251, 1994.

C. M. McDonald and C. A. Floudas. Global optimization for the phase and chemical equilibrium problem: application to the NRTL equation. *Computers and Chemical Engineering*, 19(11):1111–1141, 1995a.

C. M. McDonald and C. A. Floudas. Global optimization for the phase stability problem. *AIChE Journal*, 41(7):1798–1814, 1995b.

C. M. McDonald and C. A. Floudas. GLOPEQ: a new computational tool for the phase and chemical equilibrium problem. *Computers and Chemical Engineering*, 21:1–23, 1997.

C.M. McDonald and C.A. Floudas. Global Optimization and Analysis for the Gibbs Free Energy Function Using the UNIFAC, Wilson and ASOG Equations. *IE&C Research*, 34:1674, 1995c.

K. McKinnon and M. Mongeau. A Generic Global Optimization Algorithm for the Chemical and Phase Equilibrium Problem. *J. Global Optim.*, 12:325–351, 1998.

K. I. M. McKinnon, C. Millar, and M. Mongeau. Global Optimization for the Chemical and Phase Equilibrium Problem using Interval Analysis. In *Proceedings of the State of the Art in Global Optimization: Computational Methods and Applications (Eds. C.A. Floudas and P.M. Pardalos)*, Kluwer Book Series on Nonconvex Optimization and Its Applications. Kluwer Academic Publishers, 1996.

E. C. Meng, B. K. Shoichet, and I. D. Kuntz. Automated docking with grid-based energy evaluation. *J. Comput. Chem.*, 13:505–524, 1992.

N. Metropolis, A.W. Rosenbluth, A.H. Teller, and E. Telleri. Equation of State Calculations by Fast Computing Machines. *J. Chem. Phys.*, 21:1087, 1953.

C.A. Meyer and C.L.E. Swartz. A Regional Convexity Test for Global Optimization: Application to the Phase Equilibrium Problem. *Computers & Chemical Engineering*, 22:1407–1418, 1998.

R. Meyer. The Validity of a Family of Optimization Methods. *SIAM Journal on Control*, 8:41, 1970.

M. L. Michelsen. The Isothermal Flash Problem: Part I. Stability. *Fluid Phase Equilibria*, 9:1–19, 1981.

A. Migdalas, P. M. Pardalos, and P. Varbrand. *Multilevel Optimization : Algorithms and Applications.* Nonconvex Optimization and its Applications. Kluwer Academic Publishers, 1997.

M. D. Miller, S. K. Kearsley, D. J. Underwood, and R. P. Sheridan. FLOG : A system to select "quasi-flexible" ligand complementary to a receptor of known three-dimensional structure. *J. Comput. Aided Mol. Design*, 8:153–174, 1994.

A. Miranker and M. Karplus. Functionality maps of binding sites: a multicopy simultaneous search method. *Proteins*, 1:29–34, 1991.

E. S. Mistakidis and G. E. Stavroulakis. *Nonconvex Optimization in Mechanics : Algorithms, Heuristics and Engineering Applications by the F.E.M.* Nonconvex Optimization and its Applications. Kluwer Academic Publishers, 1997.

S. Miyamoto and P. A. Kollman. What determines the strength of noncovalent association of ligands to proteins in aqueous solutions ? *Proc. Natl. Acad. Sci. USA.*, 90:8402–8406, 1993.

J. Mockus, W. Eddy, A. Mockus, L. Mockus, and G. V. Reklaitis. *Bayesian Heuristic Approach to Discrete and Global Optimization : Algorithms, Visualization, Software and Applications.* Nonconvex Optimization and its Applications. Kluwer Academic Publishers, 1996.

L. Mockus and G.V. Reklaitis. A New Global Optimization Algorithm for Batch Process Scheduling. In C.A. Floudas and P.M. Pardalos, editors, *State of the Art In Global Optimization*, pages 521–538. Kluwer Academic Publishers, 1996.

M. J. Mohideen, J. D. Perkins, and E. N. Pistikopoulos. Optimal design of dynamic systems under uncertainty. *AIChE Journal*, 42(8):2251–2272, 1996.

M. J. Mohideen, J. D. Perkins, and E. N. Pistikopoulos. Robust Stability Considerations in Optimal Design of Dynamic Systems Under Uncertainty. *Journal of Process Control*, 7(5):371–385, 1997.

F. A. Momany, L. M. Carruthers, R. F. McGuire, and H. A. Scheraga. Intermolecular potentials from crystal data. III. Determination of empirical potentials and application to the packing configurations and lattice energies in crystals of hydrocarbons amines. *J. Phys. Chem.*, 78:1595–1620, 1974.

F. A. Momany, L. M. Carruthers, and H. A. Scheraga. Intermolecular potentials from crystal data. IV. Application of empirical potentials to the packing configurations and lattice energies in crystals of amino acids. *J. Phys. Chem.*, 78:1621–1630, 1974.

F. A. Momany, R. F. McGuire, A. W. Burgess, and H. A. Scheraga. Energy parameters in polypeptides. VII. Geometric parameters nonbonded interactions and intrinsic torsional potentials for the naturally occurring amino acids. *J. Phys. Chem.*, 79:2361–2381, 1975.

D. Monos, A. Soulika, E. Argyris, J. Gorga, L. Stern, V. Magafa, P. Cordopatis, I.P. Androulakis, and C.A. Floudas. *Proceedings of 12th International Histocompatibility World Conference HLA diversity:functional and medical implications.* D.Charron Ed., 1997.

J. B. Moon and J. Howe. Computer design of bioactive molecules: a method for receptor-based de novo ligand design. *Proteins*, 11:314–328, 1991.

R. Moore, E. Hansen, and A. Leclerc. Rigorous Methods for Global Optimization. In C. A. Floudas and P. M. Pardalos, editors, *Recent Advances in Global Optimization*, Princeton Series in Computer Science, page 321. Princeton University Press, 1992.

R. E. Moore. *Interval Analysis.* Prentice-Hall, Englewood Cliffs, New Jersey, 1966.

R. E. Moore. *Methods and applications of interval analysis.* SIAM, Philadelphia, 1979.

L. B. Morales, R. Gardu no Juárez, and D. Romero. Applications of simulated annealing to the multiple-minima problem in small peptides. *J. Biomol. Struct. Dyn.*, 8:721–735, 1991.

L. B. Morales, R. Gardu no Juárez, and D. Romero. The multiple-minima problem in small peptides revisited. The threshold accepting approach. *J. Biomol. Struct. Dyn.*, 9:951–957, 1992.

J.J. Moré. The Levenberg–Marquardt algorithm: implementation and theory. In *Numerical Analysis–Proceedings of the Biennial Conference, Lecture Notes in Mathematics*, number 630, page 105, Berlin, Germany, 1987. Springer–Verlag.

A.P. Morgan. *Solving Polynomial Systems Using Continuation for Engineering and Scientific Problems.* Pentice–Hall, Inc., 1987.

T. Mori and H. Kokame. Eigenvalue bounds for a certain class of interval matrices. *IEICE Trans. Fundamentals*, E77-A(10):1707–1709, 1994.

B. A. Murtagh and M. A. Saunders. *MINOS 5.4 User's Guide.* Systems Optimization Laboratory, Dept. of Operations Research, Stanford University, CA., 1983.

K. G. Murty and S. N. Kabadi. Some NP-complete problems in quadratic and nonlinear programming. *Mathematical Programming*, 39:117, 1987.

I.M. Navon, F.B. Brown, and H. Robertson. A Combined Simulated Annealing and Quasi-Newton like Conjugate-Gradient Method for Determining the Structure of Mixed Argon-Xenon Clusters. *Computers in Chemistry*, 14:305, 1990.

A. Nayeem, J. Vila, and H. A. Scheraga. A comparative study of the simulated annealing and Monte Carlo with minimization approaches to the the minimum energy structures of polypeptides : [met]-enkephalin. *J. Comput. Chem.*, 12:594–605, 1991.

G. Némethy, K. D. Gibson, K. A. Palmer, C. N. Yoon, G. Paterlini, A. Zagari, S. Rumsey, and H. A. Scheraga. Energy parameters in polypeptides. 10. Improved geometrical parameters and nonbonded interactions for use in

the ECEPP/3 algorithm with application to proline-containing peptides. *J. Phys. Chem.*, 96:6472–6484, 1992.

G. Némethy, M. S. Pottle, and H. A. Scheraga. Energy parameters in polypeptides. 9. Updating of geometrical parameters and hydrogen bond interactions for the naturally occurring amino acids. *J. Phys. Chem.*, 87:1883–1887, 1983.

G. L. Nemhauser and L.A. Wolsey. *Integer and Combinatorial Optimization.* Interscience Series in Discrete Mathematics and Optimization. John Wiley & Sons, 1988.

A. Neumaier. *Interval Methods for Systems of Equations.* Encyclopedia of Mathematics and its Applications. Cambridge University Press, 1990.

A. Neumaier. An optimality criterion for global quadratic optimization. *Journal of Global Optimization*, 2:201–208, 1992.

A. Neumaier. Molecular modeling of proteins and mathematical prediction of protein structure. *SIAM Review*, 39(3):407–460, 1997b.

J.A. Northby. Structure and Binding of Lennard-Jones Clusters : $13 < N < 147$. *J. Chem. Phys.*, 87:6166, 1987.

Z. Novak, Z. Kravanja, and I.E. Grossmann. Simultaneous Synthesis of Distillation Sequences in Overall Process Schemes Using an Improved MINLP Approach. *Comput. chem. engng.*, 20(12):1425–1440, 1996.

O. Odele and S. Macchietto. Computer aided molecular design: A novel method for optimal solvent selection. *Fluid Phase Equilibria*, 82:47, 1993.

K.A. Olszewski, L. Piela, and H.A. Scheraga. Mean Field Theory as a Tool for Intramolecular Conformational Optimization. Tests on Terminally-Blocked Alanine and Met-enkephalin. *J. Phys. Chem.*, 96:4672, 1992.

C. M. Oshiro, I. D. Kuntz, and J. S. Dixon. Flexible ligand docking using a genetic algorithm. *J. Comput. Aided Mol. Design*, 9:113–130, 1995.

G. M. Ostrovsky, M. G. Ostrovsky, and G. W. Mikhailow. Discrete optimization of chemical processes. *Computers and Chemical Engineering*, 14:111–117, 1990.

J. Outrata, M. Kocvara, and J. Zowe. *Nonsmooth Approach to Optimization Problems with Equilibrium Constraints: Theory, Applications and Numerical results.* Nonconvex Optimization and its Applications. Kluwer Academic Publishers, 1998.

F. Palacios-Gomez, L.S. Lasdon, and M. Engquist. Nonlinear Optimization by Successive Linear Programming. *Mgmt. Sci.*, 28(10):1106, 1982.

J.R. Paloschi and J.D. Perkins. An Implementation of Quasi-Newton Methods for Solving Sets of Nonlinear Equations. *Comput. chem. engng.*, 12:767, 1988.

S. Papageorgaki and G.V. Reklaitis. Optimal design of multipurpose batch plants: 1. Problem formulation. *I&EC Res.*, 29(10):2054, 1990.

S. Papageorgaki and G.V. Reklaitis. Optimal design of multipurpose batch plants: 2. A decomposition solution strategy. *I&EC Res.*, 29(10):2062, 1990.

K.P. Papalexandri and E.N Pistikopoulos. An MINLP retrofit approach for improving the flexibility of heat exchanger networks. *Annals of Operations Research*, 42:119, 1993.

K.P. Papalexandri and E.N. Pistikopoulos. Synthesis and Retrofit Design of Operable Heat Exchanger Networks : 1. Flexibility and Structural Controllability Aspects. *I&EC Res.*, 33:1718, 1994a.

K.P. Papalexandri and E.N. Pistikopoulos. Synthesis and Retrofit Design of Operable Heat Exchanger Networks : 2. Dynamics and Control Structure Considerations. *I&EC Res.*, 33:1738, 1994b.

K.P. Papalexandri and E.N. Pistikopoulos. Generalized Modular Representation Framework for Process Synthesis. *AIChE J.*, 42:1010, 1996.

K.P. Papalexandri, E.N. Pistikopoulos, and C.A. Floudas. Mass Exchange Networks for Waste Minimization : A Simultaneous Approach. *Chem. Eng. Res. Dev.*, 72:279, 1994.

P. M. Pardalos, X. Liu, and G. L. Xue. Protein conformation of a lattice model using tabu search. *J. Global Optim.*, 11:55–68, 1997.

P. M. Pardalos and J. B. Rosen. *Constrained global optimization: Algorithms and applications*, volume 268 of *Lecture Notes in Computer Science*. Springer Verlag, Berlin, Germany, 1987.

P. M. Pardalos and G. Schnitger. Checking local optimality in constrained quadratic programming is NP-Hard. *Oper. Res. Lett.*, 7(1):33, 1988.

P. M. Pardalos, D. Shalloway, and G. Xue. Optimization methods for computing global minima of nonconvex potential energy functions. *J. Global Optim.*, 4:117–133, 1994.

P. M. Pardalos, D. Shalloway, and G. Xue. *Global minimization of nonconvex energy functions : Molecular conformation and protein folding*, volume 23 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*. American Mathematical Society, 1996.

P. M. Pardalos, Y. Siskos, and C. Zopounidis. *Advances in Multicriteria Analysis*. Nonconvex Optimization and its Applications. Kluwer Academic Publishers, 1995.

P. M. Pardalos and S. A. Vavasis. Quadratic programming with one negative eigenvalue is NP-hard. *J. Global Optim.*, 1:15, 1991.

P.M. Pardalos and J.B. Rosen. Global optimization approach to the linear complementarity problem. *SIAM J. Scient. Stat. Computing*, 9(2):341, 1988.

U. Passy and D.J. Wilde. Generalized Polynomial Optimizations. *SIAM J. Appl. Math.*, 15:1344, 1967a.

U. Passy and D.J. Wilde. A Geometric Programming Algorithm for Solving Chemical Equilibrium Problems. *SIAM Review*, 11:89, 1967b.

H. Patino-Leal and P. M. Reilly. Statistical Estimation of Parameters in Vapor-Liquid Equilbrium. *AIChE J.*, 28(4):580, 1982.

W.E. Paul. *Fundamental Immunology*. Raven Press, 1993.

G. E. Paules, IV and C. A. Floudas. Synthesis of flexible distillation sequences for multiperiod operation. *Comput. chem. engng.*, 12(4):267, 1988.

G. E. Paules, IV and C. A. Floudas. APROS: Algorithmic development methodology for discrete-continuous optimization problems. *Oper. Res.*, 37(6):902, 1989.

G. E. Paules, IV and C. A. Floudas. Stochastic programming in process synthesis: A two-stage model with MINLP recourse for multiperiod heat-integrated distillation sequences. *Comput. chem. engng.*, 16(3):189, 1992.

F.D. Penteado and A.R. Ciric. An MINLP Approach for Safe Process Plant Layout. *I&EC Res.*, 35(4):1354–1361, 1996.

G. Perrot, B. Cheng, K. D. Gibson, K. A. Palmer J. Vila, A. Nayeem, B. Maigret, and H. A. Scheraga. MSEED: A Program for the Rapid Analytical Determination of Accessible Surface Areas and their Derivatives. *J. Comp. Chem*, 13:1–11, 1992.

S.B. Petkov and C.D. Maranas. Quantitative Assessment of Uncertainty in the Optimization of Metabolic Pathways. *Biotechnology and Bioengineering*, 56(2):145–161, 1997a.

S.B. Petkov and C.D. Maranas. Multiperiod Planning and Scheduling of Multiproduct Batch Plants Under Demand Uncertainty. *I&EC Res.*, 36(11):4864–4881, 1997b.

S.B. Petkov and C.D. Maranas. Design of Single Product Campaign Batch Plants Under Demand Uncertainty. *AIChE J.*, 44(4):896–911, 1998.

A.T. Phillips and J.B. Rosen. A parallel algorithm for constrained concave quadratic global minimization. *Mathematical Programming*, 42:421, 1988.

L. Piela, J. Kostrowicki, and H.A. Scheraga. The Multiple-Minima Problem in the Conformational Analysis of Molecules : Deformation of the Potential Energy Hypersurface by the Diffusion Equation Method. *J. Phys. Chem.*, 93:3339, 1989.

J. D. Pinter. *Global Optimization In Action - Continuous and Lipschitz Optimization.* Nonconvex Optimization and its Applications. Kluwer Academic Publishers, 1995.

E.N. Pistikopoulos and M.G. Ierapetritou. Novel Approach for Optimal Process Design Under Uncertainty. *Computers & Chemical Engineering*, 19(10):1089–1110, 1995.

E.E. Polymeropoulos and J. Brickmann. On the Origin of the Occurence of Magic Numbers in Cluster Size Distributions of Xenon in the Compressed Gas Phase. *Chem. Phys. Lett.*, 96:73, 1983.

E.E. Polymeropoulos and J. Brickmann. Magic Numbers in Ionized Rare Gas Clusters. *Surf. Sci.*, 156:563, 1985.

R. Pörn, I. Harjunkoski, and T. Westerlund. Convexification of different classes of non-convex MINLP problems. Submitted to *Computers and Chemical Engineering*, 1997.

M.J.D. Powell. *A new algorithm for unconstrained optimization: Nonlinear programming.* Academic Press, New York, 1970.

P. Psarris and C. A. Floudas. Robust Stability Analysis of Systems with Real Parameter Uncertainty : A Global Optimization Approach. *International Journal of Robust and Nonlinear Control*, 6:699–717, 1995.

I. Quesada and I. E. Grossmann. An LP/NLP based branch and bound algorithm for convex MINLP optimization problems. *Computers and Chemical Engineering*, 16:937–947, 1992.

I. Quesada and I. E. Grossmann. Global Optimization Algorithm for Heat Exchanger Networks. *I&EC Res.*, 32:487, 1993.

I. Quesada and I. E. Grossmann. A Global Optimization Algorithm Linear Fractional and Bilinear Programs. *J. Global Optim.*, 6:39–76, 1995.

I. Quesada and I. E. Grossmann. Alternative Bounding Approximations for the Global Optimization of Various Engineering Design Problems. In I. E. Grossmann, editor, *Global Optimization in Engineering Design*, Kluwer Book Series in Nonconvex Optimization and its Applications, pages 309–332, 1996. Chapter 10.

L. B. Rall. *Automatic Differentiation: Techniques and Applications.* Lecture Notes in Computer Science. Springer-Verlag, 1981.

G. N. Ramachandran and V. Saisekharan. Conformation of Polypeptides and Proteins. *Advances in Protein Chemistry*, 23:283–437, 1968.

V.S. Raman and C.D. Maranas. Optimization in Product Design with Properties Correlated with Topological Indices. *Comput. chem. engng.*, 45:997–1017, 1999.

M. Ratner, L.S. Lasdon, and A. Jain. Solving Geometric Problems Using GRG : Results and Comparisons. *J. Opt. Th. Appl.*, 26:253, 1978.

H. Ratschek and J. Rokne. *Computer Methods for the Range of Functions.* Ellis Horwood Series in Mathematics and its Applications. Halsted Press, 1984.

H. Ratschek and J. Rokne. *New Computer Methods for Global Optimization.* Halsted Press, Chichester, Great Britain, 1988.

H. Ratschek and J. Rokne. Experiments using Interval Analysis for Problem Solving a Circuit Design Problem. *J. Global Opt.*, 3:501. 1993.

H.J. Reinhart and D.W.T. Rippin. The Design of Flexible Batch Chemical Plants. Presented at the 1986 AIChE Annual Meeting, 1986.

H.J. Reinhart and D.W.T. Rippin. Design of Flexible Multi-Product Plants: A New Procedure for Optimal Equipment Sizing Under Uncertainty. Presented at the 1987 AIChE Annual Meeting, 1987.

G.V. Reklaitis. Perspectives on scheduling and planning process operations. In *Proc. 4th. Int. Symp. on Process Systems Engineering, Montreal, Canada*, 1991.

G.V. Reklaitis and K.M. Ragsdell. *Engineering Optimization.* John Wiley & Sons, New York, NY, 1983.

H. Renon and J. M. Prausnitz. Local Compositions in Thermodynamic Excess Functions for Liquid Mixtures. *AIChE J.*, 14(1):135, 1968.

C. A. Reynolds, P. M. King, and W. G. Richards. Free energy calculations in molecular biophysics. *Mol. Phys.*, 76:251–275, 1992.

N. L. Ricker. Comparison of Methods for Nonlinear Parameter Estimation. *Industrial & Engineering Chemistry Process Design and Development*, 23(2):283–286, 1984.

M.J. Rijckaert and X.M. Martens. Biographical Note On Geometric Programming. *J. Opt. Th. Appl.*, 2:325, 1978a.

M.J. Rijckaert and X.M. Martens. Comparison of Generalized Geometric Programming Algorithms. *J. Opt. Th. Appl.*, 26:205, 1978b.

D. R. Ripoll and H. A. Scheraga. The multiple minima problem in the conformational analysis of polypeptides. III. An electrostatically driven Monte Carlo method – tests on enkephalin. *J. Protein Chem.*, 8:263–287, 1989.

A. W. Roberts and D. E. Varberg. *Convex Functions.* Academic Press, 1973.

V. Rod and V. Hancil. Iterative Estimation of Model Parameters when Measurements of All Variables are Subject to Error. *Comput. chem. engng.*, 4:33, 1980.

J. Rohn. Bounds on Eigenvalues of Interval Matrices. *Z. Angew. Math. Mech*, 78:S1049–S1050, 1998.

R. Rosenfeld, Q. Zheng, S. Vajda, and C. DeLisi. Computing the structure of bound peptides. Application to antigen recognition by class I major histocompatibility complex receptors. *J. Mol. Biol.*, 234:515–521, 1993.

S. H. Rotstein and M. A. Murcko. GensStar: a program for de novo drug design. *J. Comput. Aided Mol. Design*, 7:23–43, 1993.

R. Rouhani, W. Lasdon, L. Lebow, and A.D. Warren. A generalized Benders decomposition approach to reactive source planning in power systems. *Mathematical Programming Study*, 25:62, 1985.

H. S. Ryoo and N. V. Sahinidis. Global optimization of nonconvex NLPs and MINLPs with applications in process design. *Computers and Chemical Engineering*, 19(5):551–566, 1995.

H. S. Ryoo and N. V. Sahinidis. A Branch and Reduce Approach to Global Optimization. *J. Global Optim.*, 8(2):107–139, 1996.

N.V. Sahinidis and I.E. Grossmann. Convergence properties of generalized Benders decomposition. *Comput. chem. engng.*, 15(7):481, 1991.

A. Salgovic, V. Hlavacek, and V. Ilavsky. Global Simulation of Countercurrent Separation Processes via One Parameter Imbedding Techniques. *Chem. Eng. Sci.*, 36:1599, 1981.

H.E. Salomone and O.A. Iribarren. Posynomial Modelling of Batch Plants : A Procedure to Include Porcess Decision Variables. *Comput. chem. engng.*, 16:173, 1992.

H. A. Scheraga. Predicting Three–Dimensional Structures of Oligopeptides. In K. B. Lipkowitz and D. B. Boyd, editors, *Reviews in Computational Chemistry*, volume 3, pages 73–142. VCH Publishers, 1992.

H.A. Scheraga. *PACK: Programs for Packing Polypeptide Chains*, 1996. online documentation.

C.A. Schnepper and M.A. Stadtherr. Robust Process Simulation using Interval Methods. *Comput. chem. engng.*, 20:187–199, 1996.

C. A. Schweiger and C. A. Floudas. Interaction of design and control: optimization with dynamic models. In W. W. Hager and P. M. Pardalos, editors, *Optimal Control: Theory, Algorithms, and Applications*, pages 388–435. Kluwer Academic Publishers, 1998b.

C. A. Schweiger and C. A. Floudas. Optimization Framework for the Synthesis of Complex Reactor Networks. *I&EC Res.*, 1998c.

C.A. Schweiger and C. A. Floudas. *MINOPT: a Software Package for Mixed-Integer Nonlinear Optimization, User's Guide.* Computer-Aided Systems Laboratory, Dept. of Chemical Engineering, Princeton University, NJ, 1997.

H. Schwetlick and V. Tiller. Numerical Methods for Estimating Parameters in Nonlinear Models with Errors in the Variables. *Technometrics*, 27(1):17, 1985.

J.D. Seader and R. Kumar. Multiple Solutions in Process Simulation. In *ASPENWORLD 94*, number November 6–9, Boston, MA, 1994.

J.D. Seader, M. Kuno, W.-J. Lin, S.A. Johnson, K. Unsworth, and J.W. Wiskin. Mapped Continuation Methods for Computing All Solutions to General Systems of Nonlinear Equations. *Comput. chem. engng.*, 14(1):71, 1990.

N. Shah and C.C. Pantelides. Design of Multipurpose Batch Plants with Uncertain Production Requirements. *Ind. Eng. Chem. Res.*, 31:1325–1337, 1992.

D. Shalloway. Packet Annealing : A Deterministic Method for Global Minimization : Application to Molecular Conformation. *J. Global Opt.*, 3:281, 1992.

J. P. Shectman and N. V. Sahinidis. A Finite Algorithm for Global Minimization of Separable Concave Programs. In *Proceedings of the State of the Art in Global Optimization: Computational Methods and Applications (Eds. C.A. Floudas and P.M. Pardalos)*, Kluwer Academic Series on Nonconvex Optimization and Its Applications, 1996.

J. P. Shectman and N. V. Sahinidis. A Finite Algotithm for Global Optimization of Separable Concave Functions. *J. Global Optim.*, 12:1–36, 1998.

J. Shen and F. A. Quiocho. Calculation of binding energy differences for receptor-ligand systems using the Poisson-Boltzmann methods. *J. Comput. Chem.*, 16:445–448, 1995.

H. Sherali. Global Optimization of Nonconvex Polynomial Programming Problems Having Rational Exponents. *J. Global Optim.*, 1998.

H. Sherali and C. H. Tuncbilek. A Global Optimization Algorithm for Polynomial Programming Problems Using a Reformulation-Linearization Technique. *J. Global Optim.*, 2:101, 1992.

H. Sherali and C. H. Tuncbilek. Tight Reformulation-Linearization Technique Representations for Solving Nonconvex Quadratic Programming Problems. *Submitted for Publication*, 1994.

H. D. Sherali and A. Alameddine. A new reformulation-linearization technique for bilinear programming problems. *Journal of Global Optimization*, 2:379–410, 1992.

K. Shimizu and E. Aiyoshi. A New Computational Method for Stackelberg and Min-Max Problems by Use of a Penalty Method. *IEEE Trans. on Aut. Control*, AC-26:460, 1981.

J. Shin and M. Jhon. High Directional Monte Carlo Procedure Coupled with the Temperature and Annealing Method to Obtain the Global Energy Minimum Structure of Polypeptides and Proteins. *Biopolymers*, 31:177–185, 1991.

B. K. Shoichet and I. D. Kuntz. Protein docking and complementarity. *J. Mol. Biol.*, 221:327–346, 1991.

N. Z. Shor. *Nondifferentiable Optimization and Polynomial Problems.* Nonconvex Optimization and its Applications. Kluwer Academic Publishers, 1998.

A. Sideris and Sànchez. Fast Computation of the Multivariable Stability Margin for Real Interrelated Uncertain Parameters. *in Proc. ACC, Atlanta, GA*, 1988.

M. L. Silver, H. C. Guo, J. L. Strominger, and D. Wiley. Atomic structure of a human MHC molecule presenting an influenza virus peptide. *Nature*, 360:367–368, 1992.

H. Skrifvars, I. Harjunkoski, T. Westerlund, Z. Kravanja, and R. Pörn. Comparison of different MINLP methods applied on certain chemical engineering problems. *Computers and Chemical Engineering Supplement*, 20:S333–S338, 1996.

E.M.B. Smith and C.C. Pantelides. Global optimisation of general process models. In I.E. Grossmann, editor, *Global Optimization in Engineering Design*, pages 355–386. Kluwer Academic Publishers, 1996.

E.M.B. Smith and C.C. Pantelides. Global optimisation of nonconvex MINLPs. *Computers and Chemical Engineering Supplement*, 21:S791–S796, 1997.

E.M.B. Smith and C.C. Pantelides. A Symbolic Reformulation/Spatial Branch and Bound Algorithm for the Global Optimization of nonconvex MINLPs. *Computers and Chemical Engineering*, 23:457–478, 1999.

J.M. Smith. *Chemical Engineering Kinetics.* Butterworth, Stoneham, MA, 1985.

J.V. Smith, R.W. Missen, and W.R. Smith. General optimality criteria for multiphase multireaction chemical equilibrium. *AIChE J.*, 39(4):707, 1993.

W. H. Southwell. Fitting Expermental Data. *J. of Comp. Physics*, 4:465, 1969.

R.E. Sparrow, G.J. Forder, and D.W.T. Rippin. The Choice of Equipment Sizes for Multiproduct Batch Plants. Heuristics vs. Branch and Bound. *Ind. Eng. Chem. Proc. Des. Dev.*, 14:197–203, 1975.

G. R. Sriniwas and Y. Arkun. A Global Solution to the Nonlinear Model Predictive Control Algorithms Using Polynomial ARX Models. *Comp. & Chem. Eng.*, 21:431–439, 1997.

G. H. Staus, L. T. Biegler, and B. E. Ydstie. Adaptive Control via Nonconvex Optimization. In *Proceedings of the State of the Art in Global Optimization: Computational Methods and Applications (Eds. C.A. Floudas and P.M. Pardalos)*, Kluwer Academic Series on Nonconvex Optimization and Its Applications, 1996.

S.K. Stefanis, A.G. Livingston, and E.N. Pistikopoulos. Environmental Impact Considerations in the Optimal Design and Scheduling of Batch Processes. *Computers & Chemical Engineering*, 21(10):1073–1094, 1997.

G. Stephanopoulos and A.W. Westerberg. The Use of Hestenes' Method of Multipliers to Resolve Dual Gaps In Engineering System Optimization. *JOTA*, 15:285, 1975.

C. Stephens. Interval and bounding Hessians. In I. M. et al. Bomze, editor, *Developments in Global Optimization*, pages 109–199. Kluwer Academic Publishers, 1997.

L. Stern, J. Brown, T. Jardetsky, J. Gorga, R. Urban, L. Strominger, and D. Wiley. Crystal structure of the human class II MHC protein HLA-DR1 complexed with an influenza virus peptide. *Nature*, 368:215–221, 1994.

L. Stern, J. Brown, T. Jardetzky, J. Gorga, R. Urban, L. Strominger, and D. Wiley. Crystal structure of the human class II MHC protein HLA-DR1 complexed with an influenza virus peptide. *Nature*, 368:215–221, 1994.

W. E. Stewart, M. Caracotsios, and J. P. Sorensen. Parameter Estimation from Multiresponse Data. *AIChE J.*, 38(5):641, 1992.

D.A. Straub and I.E. Grossmann. Evaluation and Optimization of Stochastic Flexibility in Multiproduct Batch Plants. *Comput. chem. engng.*, 16:69–87, 1992.

D.R. Stull, E.F. Westrum, and G.C. Sinke. *The Chemical Thermodynamics of Organic Compounds.* J. Wiley, New York, 1969.

S. Subrahmanyam, J.F. Pekny, and G.V. Reklaitis. Design of Batch Chemical Plants Under Market Uncertainty. *Ind. Eng. Chem. Res.*, 33:2688–2701, 1994.

R. Susnow, R.B. Nachbar, C. Schutt, and H. Rabitz. Sensitivity of Molecular Structure to Interatomic Potentials. *J. Phys. Chem.*, 95:8585, 1991.

R.E. Swaney. Global Solution of Algebraic Nonlinear Programs. *Presented at Annual AIChE Meeting, Chicago*, 1990.

A. S. Teja and J. S. Rowlinson. The Prediction of the Thermodynamic Properties of Fluids and Fluid Mixtures - IV. Critical and Azeotropic States. *Chemical Engineering Science*, 28:529–538, 1973.

H.B. Thompson. Calculation of Cartesian Coordinates and their Derivatives from Internal Molecular Coordinates. *J. Chem. Phys.*, 47:3407, 1969.

T. B. Tjoa and L. T. Biegler. Reduced Successive Quadratic Programming Strategy for Errors-in-Variables Estimation. *Comput. chem. engng.*, 16(6):523, 1992.

A. Torn and A. Zilinskas. *Global Optimization*, volume 350 of *Lecture Notes in Computer Science.* Springer-Verlag, Berlin, Germany, 1989.

J.M. Troyer and F.E. Cohen. *Simplified models for understanding and predicting protein structure.* VCH Publishers, New York, NY, 1991.

A. G. Tsirukis and G. V. Reklaitis. Feature Extraction Algorithms for Constrained Global Optimization : Parts I and II. *Annals of Operations Research*, 42:229–275, 1993.

M. Turkay and I.E. Grossmann. Logic-based MINLP Algorithms for the Optimal Synthesis of Process Networks. *Comput. chem. engng.*, 20(8):959–978, 1996.

H. Tuy. *Convex Analysis and Global Optimization.* Nonconvex Optimization and its Applications. Kluwer Academic Publishers, 1998.

H. Tuy, A. Migdalas, and P. Varbrand. A Global Optimization Approach for the Linear Two-Level Program. *J. Global Opt.*, 3:1, 1993a.

H. Tuy, A. Migdalas, and P. Varbrand. A Quasiconcave Minimization Method for Solving Linear Two-Level Programs. *J. Global Opt.*, 4:243, 1993b.

R. G. Urban and R. M. Chicz. *MHC Molecules: Expression, Assembly and Function.* R. G. Landes Company and Chapman & Hall, 1996.

R. Vaidyanathan and M. El-Halwagi. Global optimization of nonconvex nonlinear programs via interval analysis. *Computers and Chemical Engineering*, 18:889–897, 1994.

R. Vaidyanathan and M. El-Halwagi. Global optimization of nonconvex MINLP's by interval analysis. In I. E. Grossmann, editor, *Global Optimization in Engineering Design*, pages 175–193. Kluwer Academic Publishers, 1996.

S. Vaidyaraman and C.D. Maranas. Optimal Synthesis of Refrigation Cycles and Selection of Refrigerants. *AIChE J.*, 45:997–1017, 1999.

S. Vajda, M. Sippl, and J. Novotny. Empirical potentials and functions for protein folding and binding. *Curr. Opinion Struct. Biol.*, 7:222–228, 1997.

P. Valko and S. Vajda. An Extended Marquardt-type Procedure for Fitting Error-in-Variables Models. *Comput. chem. engng.*, 11(1):37, 1987.

W. F. van Gunsteren and H. J. C. Berendsen. *GROMOS.* Groningen Molecular Simulation, Groningen, The Netherlands, 1987.

J. G. VanAntwerp, R. D. Braatz, and N. V. Sahinidis. Globally Optimal Robust Control for Systems with Time-Varying Nonlinear Perturbations. *Comp. & Chem. Eng.*, 21:S125–130, 1997.

L. Vandenberghe and S. Boyd. Semidefinite programming. *SIAM Review*, 38:49–95, 1996.

J. Vaselenak, I.E. Grossmann, and A.W. Westerberg. An embedding formulation for the optimal scheduling and design of multiproduct batch plants. *I&EC Res.*, 26(1):139, 1987.

J. Vaselenak, I.E. Grossmann, and A.W. Westerberg. Optimal retrofit design of multipurpose batch plants. *I&EC Res.*, 26(4):718, 1987.

M. Vásquez, G. Némethy, and H. A. Scheraga. Computed Conformational States of the 20 Naturally Occurring Amino Acid Residues and of the Prototype Residue α–Aminobutyric Acid. *Macromolecules*, 16:1043–1049, 1983.

M. Vásquez, G. Némethy, and H. A. Scheraga. Conformational energy calculations on polypeptides and proteins. *Chem. Rev.*, 94:2183–2239, 1994.

M. Vásquez, G. Némethy, and H. A. Scheraga. Conformational Energy Calculations on Polypeptides and Proteins. *Chemical Reviews*, 94:2183–2239, 1994.

S. Vavasis. *Nonlinear Optimization: Complexity Issues.* Oxford University Press, New York, N.Y., 1991.

A. Vecchietti and I.E. Grossmann. LOGMIP : A Disjunctive 0-1 Nonlinear Optimizer for Process System Models. *Comput. chem. engng.*, 23:555–565, 1999.

G. Verkhivker, K. Appelt, S. T. Freer, and J. E. Vilafranca. Empirical free energy calculations of ligand-protein crystallographic complexes. 1. Knowledge based ligand-protein interaction potentials applied to the prediction of human immunodeficiency virus 1 protease binding affinity. *Protein Eng.*, 8:677–691, 1995.

A. Vicino, A. Tesi, and M. Milanese. Computation of Nonconservative Stability Perturbation Bounds for Systems with Nonlinearly Correlated Uncertainties. *IEEE Trans. Autom. Contr.*, 35:835, 1990.

J. Vila, R.L. Williams, M. Vasquez, and H.A. Scheraga. Empirical Solvation Models can be used to Differentiate Native from Non-Native Conformations of Bovine Pancreatic Trypsin Inhibitor. *Proteins*, pages 199–218, 1991.

J. Viswanathan and I. E. Grossmann. A combined penalty function and Outer-Approximation for MINLP optimization. *Computers and Chemical Engineering*, 14(7):769–782, 1990.

V. Visweswaran and C. A. Floudas. A global optimization algorithm GOP for certain classes of nonconvex NLPs : II. Application of theory and test problems. *Computers and Chemical Engineering*, 14:1419–1434, 1990.

V. Visweswaran and C. A. Floudas. Unconstrained and constrained global optimization of polynomial functions in one variable. *J. Global Optim.*, 2:73, 1992.

V. Visweswaran and C. A. Floudas. New properties and computational improvement of the GOP algorithm for problems with quadratic objective function and constraints. *Journal of Global Optimization*, 3(3):439–462, 1993.

V. Visweswaran and C. A. Floudas. *cGOP: A User's Guide*. Princeton University, Princeton, New Jersey, 1995.

V. Visweswaran and C. A. Floudas. New formulations and branching strategies for the GOP algorithm. In I. E. Grossmann, editor, *Global Optimization in Engineering Design*, Kluwer Book Series in Nonconvex Optimization and its Applications, pages 75–110, 1996a. Chapter 3.

V. Visweswaran and C. A. Floudas. Computational results for an efficient implementation of the GOP algorithm and its variants. In I. E. Grossmann, editor, *Global Optimization in Engineering Design*, Kluwer Book Series in Nonconvex Optimization and its Applications, pages 110–154, 1996b. Chapter 4.

V. Visweswaran, C.A. Floudas, M.G. Ierapetritou, and E.N. Pistikopoulos. A Decomposition-Based Global Optimization Approach for Solving Bilevel Linear and Quadratic Programs. In C.A. Floudas and P.M. Pardalos, editors, *State of the Art In Global Optimization : Computational Methods and Applications*, Kluwer Book Series in Nonconvex Optimization and its Applications, pages 139–162, 1996.

D.G. Vlachos, L.D. Schmidt, and R. Aris. Structures of Small Metal Clusters : I. Low Temperature Behavior. *J. Chem. Phys.*, 96:6880, 1992a.

B. von Freyberg and W. Braun. Efficient search for all low energy conformations of polypeptides by Monte Carlo methods. *J. Comp. Chem.*, 12:1065–1076, 1991.

V.T. Voudouris and I.E. Grossmann. Mixed-Integer Linear Programming Reformulations for Batch Process Design with Discrete Equipment Sizes. *Ind. Eng. Chem. Res.*, 31:1315–1325, 1992.

A. Šali, E. Shaknovich, and M. Karplus. Thermodynamics and kinetics of protein folding. In *DIMACS Series in Discrete Mathematics and Theoretical*

Computer Science, volume 23, pages 199–213. American Mathematical Society, 1996.

A. Wallquist, R. L. Jernigan, and D. G. Covell. A preference-based free energy parameterization of enzyme-inhibitor binding. Applications to HIV-1 protease inhibitor design. *Protein Sci.*, 4:1881–1903, 1995.

F.F.Y. Walraven and P.V. van Rompay. An improved phase-splitting algorithm. *Comput. chem. engng.*, 12(8):777, 1988.

H. Wang. Grid-search molecular accessible surface algorithm for solving the protein docking problem. *J. Comput. Chem.*, 12:746–750, 1991.

S. Wang and W. B. Whiting. New Algorithm for Calculation of Azeotropes from Equations of State. *Ind. Eng. Chem. Proc. Des. Dev.*, 25:547–551, 1986.

L.T. Watson, S.C. Billips, and A.P. Morgan. Algorithm 652 HOMPACK : A Suite of Codes for Globally Convergent Homotopy Algorithms. *ACM Trans. Math. Soft.*, 13:281, 1987.

R. J. Wawak, K. D. Gibson, and H. A. Scheraga. Gradient Discontinuities in Calculations Involving Molecular Surface Area. *J. Math. Chem.*, 15:207–232, 1994.

T.L. Wayburn and J.D. Seider. Solution of Systems of Interlinked Distillation Columns by Differential Homotopy–Continuation Methods. In *Proc. 2nd Int. Conf. Found. Computer–Aided Process Design*, 1983.

S. J. Weiner, P. A. Kollman, D.A. Case, U.C. Singh, C. Ghio, G. Alagona, S. Profeta, and P. Weiner. A new force field for molecular mechanical simulation of nucleic acids and proteins. *J. Am. Chem. Soc.*, 106:765–784, 1984.

S. J. Weiner, P.A. Kollman, D.T. Nguyen, and D.A. Case. An all atom force field for simulations of proteins and nucleic acids. *Journal of Computational Chemistry*, 7(2):230–252, 1986.

H.S. Wellons and G.V. Reklaitis. The Design of Multiproduct Batch Plants Under Uncertainty With Staged Expansion. *Comput. chem. engng.*, 13:115–126, 1989.

M.C. Wellons and G.V. Reklaitis. Scheduling of multipurpose batch chemical plants: 1. Formulation of single-product campaigns. *I&EC Res.*, 30(4):671, 1991.

M.C. Wellons and G.V. Reklaitis. Scheduling of multipurpose batch chemical plants: 1. Multiple product campaign formulation and production planning. *I&EC Res.*, 30(4):688, 1991.

A. W. Westerberg and J. V. Shah. Assuring a Global Optimum by the Use of an Upper Bound on the Lower (Dual) Bound. *Comput. chem. engng.*, 2:83, 1978.

K. M. Westerberg and C. A. Floudas. Locating All Transition States and Studying the Reaction Pathways of Potential Energy Surfaces. *Journal of Chemical Physics*, 110(18):9259–9295, 1999a.

K. M. Westerberg and C. A. Floudas. Dynamics of Peptide Folding : Transition States and Reaction Pathways of Solvated and Unsolvated Tetra-Alanine. *Journal of Global Optimization*, 1999b.

T. Westerlund and F. Pettersson. An extended cutting plane method for solving convex MINLP problems. *Computers and Chemical Engineering*, 19:131–136, 1995.

T. Westerlund, F. Pettersson, and I. E. Grossmann. Optimization of pump configuration problems as a MINLP problem. *Computers and Chemical Engineering*, 18(9):845–858, 1994.

T. Westerlund, H. Skrifvars, I. Harjunkoski, and R. Pörn. An extended cutting plane method for a class of non-convex MINLP problems. *Computers & Chemical Engineering*, 22(3):357–365, 1998.

D.J. White and G. Anandalingam. A Penalty Function Approach for Solving Bi-Level Linear Programs. *J. Global Opt.*, 3:397, 1993.

W.B. White, S.M. Johnson, and G.B. Dantzig. Chemical equilibrium in complex mixtures. *J. Chem. Phys.*, 28(5):751, 1958.

S. Widagdo and W. D. Seider. Azeotropic Distillation. *AIChE Journal*, 42:96–130, 1996.

L.T. Wille. Searching Potential Energy Surfaces by Simulated Annealing. *Nature*, 34:46, 1986.

L.T. Wille. Minimum Energy Configurations of Atomic Clusters : New Results Obtained by Simulated Annealing. *Chem. Phys. Lett.*, 133:405, 1987.

L.T. Wille and J. Vennik. Computational Complexity of the Ground State Determination of Atomic Clusters. *J. Phys. A*, 18:L419, 1985.

R. L. Williams, J. Vila, G. Perrot, and H. A. Scheraga. Empirical Solvation Models in the Context of Conformational Energy Searches: Application to Bovine Pancreatic Trypsin Inhibitor. *Proteins*, 14:110–119, 1992.

G. M. Wilson. Vapor Liquid Equilibrium XI: A New Expression for the Excess Gibbs Energy of Mixing. *J. Am. Chem. Soc.*, 86:127, 1964.

D.R. Wingo. Globally minimizing polynomials without evaluating derivatives. *Intl. J. Comput. Math*, 17:287, 1985.

S. J. Wodak, M. De Crombrugghe, and J. Janin. Computer studies of interactions between macromolecules. *Prog. Biophys. Mol. Biol.*, 49:29–63, 1987.

S. J. Wodak and J. Janin. Computer analysis of protein-protein interactions. *J. Mol. Biol.*, 124:323–342, 1978.

K.W. Wucherpfennig, A. Sette, S. Southwood, C. Oseroff, M. Matsui, J.L. Strominger, and D.A. Hafler. Structural Requirements for Binding of an Immunodominant Myelin Basic Protein Peptide to DR2 Isotypes and for its Recognition by Human T Cell Clones. *Journal of Experimental Medicine*, 179:279–290, 1994.

K.W. Wucherpfennig and J.L. Strominger. Selective Binding of Self Peptides to Disease-associated MHC Molecules: A Mechanism for MHC-linked Susceptibility to Human Autoimmune Diseases. *Journal of Experimental Medicine*, 181:1597–1601, 1995.

Q. Xia and S. Macchietto. Design and Synthesis of Batch Plants : MINLP Solution Based on a Stochastic Method. *Computers & Chemical Engineering*, 21:S697–S702, 1997.

G.L. Xue. Parallel Two-Level Simulated Annealing. *J. Global Opt.*, 4:187, 1994a.

G.L. Xue. Improvement on the Northby Algorithm for Molecular Conformation : Better Solutions. *J. Global Opt.*, 4:425, 1994b.

G.L. Xue, R.S. Maier, and J.B. Rosen. Minimizing the Lennard–Jones Potential Function on a Massively Parallel Computer. In *ACM International Conference in Supercomputing*, page 409, Washington, DC., 1992.

Y. Yamada and S. Hara. Global Optimization for H-infinity Control with Constant Diagonal Scaling. *IEEE Transactions on Automatic Control*, 43:191–203, 1998.

T. F. Yee and I. E. Grossmann. Simultaneous optimization model for heat exchanger network synthesis. In I. E. Grossmann, editor, *Chemical Engineering Optimization Models with GAMS*, volume 6 of *CACHE Design Case Studies Series*. CACHE, 1991.

T.F. Yee and I.E. Grossmann. Simultaneous optimization models for heat integration - II. Heat exchanger network synthesis. *Comput. chem. engng.*, 14(10):1165–1184, 1990.

T.F. Yee, I.E. Grossmann, and Z. Kravanja. Simultaneous optimization models for heat integration - I. Area and energy targeting and modeling of multistream exchangers. *Comput. chem. engng.*, 14(10):1151–1164, 1990a.

T.F. Yee, I.E. Grossmann, and Z. Kravanja. Simultaneous optimization models for heat integration - III. Area and energy targeting and modeling of multistream exchangers. *Comput. chem. engng.*, 14(11):1185–1200, 1990b.

X. Yuan, S. Zhang, L. Pibouleau, and Domenech S. A mixed-integer nonlinear-programming method for process design. *RAIRO - Recherche Opérationnelle-Operations Research*, 22(4):331–346, 1988.

J.M. Zamora and I.E. Grossmann. A Global MINLP Optimization Algorithm for the Synthesis of Heat Exchanger Networks with no Stream Splits. *Computers & Chemical Engineering*, 22(3):367–384, 1998a.

J.M. Zamora and I.E. Grossmann. Continuous Global Optimization of Structured Process Systems Models. *Computers & Chemical Engineering*, 22(12):1749–1770, 1998b.

C. Zener. A Mathematical Aid in Optimizing Engineering Design. *Proc. Nat. Acad. Sci. U.S.A.*, 47:537, 1961.

L. Zerva, B. Cizman, N.K. Mehra, S.K. Alahari, R. Murali, C.M. Zmijewski, M. Kamoun, and D.S. Monos. Arginine at positions 13 or 70-71 of HLA-DRB1 alleles confer sucseptibility to Tuberculoid Leprosy. *Journal of Experimental Medicine*, 183:829–836, 1996.

X. Zhang and R.W.H. Sargent. The Optimal operation of mixed production facilities : General Formulation and some Solution Approaches. In *Proceedings of 5th Intl. Symp. on Process Systems Engineering*, pages 171–177, 1994.

S. S. Zimmerman, M. S. Pottle, G. Nèmethy, and H. A. Scheraga. Conformational analysis of the 20 naturally occurring amino acid residues using ECEPP. *Macromolecules*, 10:1–9, 1977.

Index

E-Matrix method, 346
H-Matrix method, 353, 360
α calculation, 343, 379, 384
αBB implementation, 657–662, 665
ϵ-global minimum, 58, 289
ϵ-global convergence, 232
(DC) transformation, 403
cGOP package, 141, 152, 162
connected variables, 134, 150, 151, 160
ϵ-global optimality, 235
qualifying constraints, 79, 150, 151
NP-class, 21, 404
NP-complete, 21, 576
NP-hard, 21, 175, 576
P-class, 20
H-**matrix**, 351
M-**matrix**, 351
bound improving, 329
consistent, 327, 328
all multiple global minima, 644
bound improving, 282, 283, 327, 329, 656, 657
condensed polynomial, 259
connected variables, 77, 79
consistent, 282, 327, 656
convex envelope, 647
diagonal shift matrix, 338
exhaustive, 282, 328
force field, 436
heterogeneous azeotrope, 669
homogeneous azeotrope, 669
least reduced axis, 379, 381
monomials, 257
partitioning, 68
posynomials, 257
primal problem, 421, 423
qualifying constraints, 203, 207, 422
reactive azeotrope, 669
selection operation, 283
signomials, 257
transformations, 68
error-in-variables, 12, 543, 544, 547
exhaustive, 328
inner relaxed dual problem, 73
interaction of synthesis, design and control, 574
inter-interactions, 489
intra-interactions, 489
local optimization, 2
molecular design, 574
multiple-minima problem, 452
primal problem, 70
process design, 573
process operations, 574
process synthesis and design under uncertainty, 574
process synthesis, 573
relaxed dual problem, 72
robust stability analysis, 392
solvation, 489
strong duality theorem, 71
GLOPEQ, 235
turing machine, 20

accessible surface area, 456
accessible volume shell, 458
activity coefficient models, 667
alkylation process design, 290, 390
all ϵ–global minima, 644
all azeotropes, 667, 669, 670
all homogeneous azeotropes, 667, 668, 692
all solutions, xvii, 3, 16, 32, 641, 643, 644, 646, 653, 670
Antoine equation, 18, 672, 680
arithmetic mean temperature difference, 605
arithmetic–geometric inequality, 259
automatic αBB termination, 630, 632
automatic differentiation, 377
azeotropes, 25

azeotropy, 669

biconvex, 3, 32, 309
bilevel linear, 173–175, 181
bilevel linear-quadratic, 187
bilevel nonlinear, 174
bilevel quadratic, 173
bilevel quadratic-quadratic, 187
bilinear, 3, 32, 309, 335, 339, 341, 380, 550, 588, 648, 673
bilinear data, 142
bilinear function, 47
bond angles, 435, 439, 448, 454
bond lengths, 435, 439, 447, 454
Born–Meyer, 10, 408
bounding operation, 327
bounds, 143
box constrained, 315, 338, 435
branch and bound, xiv, 114, 118, 334, 379, 493, 577, 578, 618, 628, 644, 653
branch and bound GOP algorithm, 120
branch-and-reduce, 579, 580
branching criteria, 526
branching methods, 550
branching strategies, 379, 603, 628
branching variables, xvii, 591, 622, 632
Buckingham, 437, 448

challenges in MINLP, 576
characteristic polynomial, 9, 297, 298, 300–304
Chen approximation, 601
closed line segment, 34
clusters, xv, 10, 24, 403, 425
comparison matrix, 351
comparison of αBB, 468
comparison to underestimation schemes, 538
comparison with interval analysis, 562, 564
complexity analysis, xv, 19, 329, 404
complexity of global optimization, 22
complexity of local optimization, 21
computational chemistry, 2, 23, 27
computational complexity, 479
concave, 673
concave function, 36, 46
concave quadratic, 386
concave quadratic problems, 164
conferences, 26
connected variables, 75
constrained nonlinear systems, 32
constrained twice differentiable NLPs, 333
continuous function, 38
continuous variable branching, 591
control data, 142
convex combination, 35
convex envelope, 45–48, 50, 52, 54, 335, 339, 588, 648, 649, 652, 673
convex function, 36
convex hull, 35
convex lower bounding, xv, 262, 318, 342, 444, 494, 517, 533, 590, 641, 646, 647, 649, 652, 653, 672, 674, 675
convex relaxation, 645, 672
convex set, 34
covariance matrix, 546
CSTR sequence design, 292
cubic lattice methods, 484
cutting plane, 578

d.c. function, 58, 63
d.c. representation, 60–63
d.c. transformation, xv, 409
data reconcilliation, 24
data structures, 142
decision problem, 20
decomposition approach, 488
design under uncertainty, 24, 25, 507, 508
deterministic global optimization, 642–644, 646
difference of convex functions, 316
difference of convex functions' transformation, 261
differentiable function, 39
differentiable quasi-concave, 44
differentiable quasi-convex, 44
discontinuous α parameters, 339
distillation systems, 25
dual problem, 71
duality theory, xiii, 70

ECEPP/3, 11, 454, 466, 491
economics, 23
engineering design, 24
epigraph, 38
equality constraints, 341
exact α calculation, 517
extended cutting plane, 577, 579, 582

facility location and allocation, 573
facility planning and scheduling, 573
feasibility approach, 577, 578
feasible solution, 57
finite ϵ-convergence, xiii, 87, 88, 646
first order constraint qualification, 68
flexible models, 484
follower, 174
fractional, 336, 341, 380, 588, 681, 685, 687
fractional trilinear, 337, 341, 588

GAMS implementation, 657–660, 663, 665
Gauss-Legendre quadrature, 512, 515
Gaussian, 10, 408
general constrained, 338
general nonconvex, 338, 380, 550, 589
generalized Benders decomposition, 576, 598, 625
generalized cross decomposition, 577
generalized geometric programming, 3, 32, 257–259, 261, 289
generalized outer approximation, 577
generalized polynomial, 650
geometric programming, 258
Gerschgorin's theorem, 344, 358
global minimum, 57
global MINLP, 571, 579
global optimality, xiii, 87, 89
global optimization, 1, 3, 18, 32
global optimization algorithm αBB, xv, 323
global optimization applications, 23
global optimization approaches, 18
GMIN-αBB, 579, 584, 617, 618, 620, 624–626, 628
grid search, 484

halfspace, 34
Haverly pooling, 159, 385
heat exchanger design, 295
heat exchanger networks, 25, 152, 155, 391, 601
Hertz's method, 350, 359
heterogeneous azeotrope, 670
homogeneous azeotrope, 671
homogeneous azeotropes, 670, 685
homotopy continuation, 642
human leukocyte antigens (HLA), 485
hybrid branch-and-bound and outer approximation, 579, 583
hybrid branching, 592
hyperplane, 34
hypograph, 38

implementation, 377, 382
indefinite quadratic, 167, 243, 247
infeasible primal problems, 82
inner relaxed dual, 76
interaction of design, synthesis and control, 573
interatomic spacings, 431
internal coordinates, 439
interval analysis, 579, 580
interval hessian, 360, 379
interval–Newton, 642, 643
iteration data, 143

Kowalik problem, 560

leader, 174
length of the input, 20
Lennard-Jones, 10, 365, 403, 404, 408, 414, 425, 431, 437
Levinthal's paradox, 452
line, 33
linear branching scheme, xiv, 134
local minimum, 57
local MINLP, 571, 576, 577
local optimization, 1, 2
locally D.C. function, 58
logic-based approaches, 577
lower bounding, 377
lower bounding hessian method, 348, 359
lower semi-continuous function, 38

magic numbers, 431
maximum likelihood, xvi, 543, 545
maximum number of iterations, 329
maximum separation, 48, 322, 335, 336, 380, 462, 646, 648, 653
measure of a matrix, 312
Mie, 10, 408
min–max, 644
minimization of maximum separation distance, 354, 360
minimization of the Gibbs free energy, 193
minimum interparticle distance, 416
minimum number of iterations, 330
MINOPT, 377, 598, 625
mixed-integer nonlinear, 3, 32, 571, 576
mixed-product campaign, xvi, 507, 530
modifications of αBB, 548
modified αBB, xvi, 507, 524
modified GOP algorithm, 181, 186
molecular biology, 2, 23, 27
molecular design, 572
molecular dynamics, 405
molecular mechanics, 436
molecules, 24, 435, 443
monotonicity analysis, 275
Monte Carlo, 405
Mori and Kokame's method, 347, 359
Morse, 10, 403, 408, 431
multiperiod tankage quality problem, 111, 162
multiple global minima, 653, 671, 672
multiproduct, 507–509, 517, 524
multipurpose, 507

necessary and sufficient conditions for convexity/concavity, 650
negative semi-definite, 41
Newton and quasi–Newton, 642
non-uniform diagonal shift, 342, 351
nonconvex function, 36

nonconvex lower bounding, 523
nonconvex MINLP, 576, 617, 624, 626
nonideal separations, 25
nonlinear constrained systems of equations, 16, 641, 646, 671
nonlinear data, 143
nonsharp separation, 391
NRTL equation, 667, 668, 679, 680, 693

optimal control, 25
optimal reactor design, 296
outer approximation, 576, 598, 625

parallel implementation, 243
parameter α, 311, 316, 317, 403, 414, 466
parameter estimation, 24, 25, 543
partitioning, 200, 230
peptide docking, 2, 25, 481, 482
phase and chemical equilibrium, 2, 24, 193
phase stability problem, 8, 228
pocket definition, 490
polyhedron, 34
polynomial-time algorithm, 20
polytope, 34, 46
pooling, 161, 243
pooling and blending, 25, 158, 251
positive semi-definite, 41, 339
potential energy, 11, 407, 436, 451, 454, 460, 491
primal problem, 73, 84, 150, 182, 201, 231
probability-based partitioning, 464
problem, 20
process control, 27
process design, 24, 27, 572
process operations, 24, 573
process synthesis, 24, 27, 572
process synthesis and design under uncertainty, 572
product of univariate functions, 50, 647, 649, 650, 676, 677, 680
propanal, 447
protein folding, 2, 24, 451
protein representation, 452, 490
pseudo-concave function, 44
pseudo-convex function, 44
pump network, 25, 624, 625

quadratic programming, 95
quadratically constrained problems, xiv, 96
quasi-concave function, 42
quasi-convex function, 42

reactor network design, 391
reactor networks, 25
reduction tests, 118
reformulation/spatial branch-and-bound, 579, 583
relaxed dual, xiv, 84, 126, 151, 177, 178, 182, 203, 229, 231, 243, 245, 318, 421, 424
respiratory mechanical model, 562
rigid models, 484
rigorous calculation of α, xv, 342
robust stability analysis, xiv, 297

scaled Gerschgorin theorem, 351, 360, 657
scoring, 482, 483
separation sequences, 162
sequential branching: binary-continuous, 591
shape based methods, 484
shape complementarity methods, 484
signomial, 3
simulated annealing, 405
single-product campaign, 509
SMIN-αBB, 579, 584, 587, 588, 595, 598–601
solvated leu–enkephalin, 476
solvated met–enkephalin, 472
solvation, 11, 451, 455, 460, 491
stability margin, 298, 300, 301
strictly concave function, 36
strictly convex function, 36
strictly quasi-concave function, 43
strictly quasi-convex function, 43
structure prediction, 403, 435, 451

tangent plane stability, 7, 193, 225
textbooks and monographs, 25
the αBB, 309, 315, 327, 329, 333, 334, 338, 339, 361, 377, 379, 382, 384, 419, 435, 444, 451, 460, 461, 464, 466, 481, 489, 507, 517, 526, 538, 543, 548, 589, 618, 620, 621, 623, 628, 657
the GOP algorithm, xiv, 67, 84, 89, 114, 118, 134, 141, 154, 158, 159, 160, 161, 164, 173, 176, 193, 200, 201, 212, 230, 231, 232, 243, 245, 247, 309, 316, 317, 403, 419, 420, 516
the GOP/MILP algorithm, 130, 131, 151
the NRTL activity coefficient, 6, 9, 193, 198, 212, 230
the GOS algorithm, 114
thermodynamics, 23
topology of transportation networks, 573
transformation of inequalities, 269
transformations, 200, 230
trilinear, 336, 341, 588
trim loss, 25, 624, 630
twice continuously differentiable, 3, 32, 62, 652

twice differentiable, 12, 40, 309, 310, 312, 315, 333, 338, 340, 343, 460, 648

underestimation strategies, xv, 334
UNIFAC equation, 667, 668, 688, 690, 691, 695
uniform diagonal shift, 342, 344, 357
UNIQUAC equation, 667, 668, 684, 685, 687, 695
unique global minimum, 57
univariate, 54, 648, 673
univariate concave, 46, 337, 380, 550, 589
univariate polynomial problems, 98
unlimited intermediate storage, 507, 532
unsolvated leu–enkephalin, 470
unsolvated met–enkephalin, 466
updates of variable bounds, xvii, 592, 622
upper bounds, 590
upper semi-continuous function, 39

valid lower bounds, xvi, 588
variable bound updates, 379, 381, 605, 629, 632
variable bounds, 549
variable scaling, 266

Wilson equation, 667, 668, 674, 675, 693

zero exclusion condition, 298

Nonconvex Optimization and Its Applications

1. D.-Z. Du and J. Sun (eds.): *Advances in Optimization and Approximation.* 1994 ISBN 0-7923-2785-3
2. R. Horst and P.M. Pardalos (eds.): *Handbook of Global Optimization.* 1995 ISBN 0-7923-3120-6
3. R. Horst, P.M. Pardalos and N.V. Thoai: *Introduction to Global Optimization* 1995 ISBN 0-7923-3556-2; Pb 0-7923-3557-0
4. D.-Z. Du and P.M. Pardalos (eds.): *Minimax and Applications.* 1995 ISBN 0-7923-3615-1
5. P.M. Pardalos, Y. Siskos and C. Zopounidis (eds.): *Advances in Multicriteria Analysis.* 1995 ISBN 0-7923-3671-2
6. J.D. Pintér: *Global Optimization in Action.* Continuous and Lipschitz Optimization: Algorithms, Implementations and Applications. 1996 ISBN 0-7923-3757-3
7. C.A. Floudas and P.M. Pardalos (eds.): *State of the Art in Global Optimization.* Computational Methods and Applications. 1996 ISBN 0-7923-3838-3
8. J.L. Higle and S. Sen: *Stochastic Decomposition.* A Statistical Method for Large Scale Stochastic Linear Programming. 1996 ISBN 0-7923-3840-5
9. I.E. Grossmann (ed.): *Global Optimization in Engineering Design.* 1996 ISBN 0-7923-3881-2
10. V.F. Dem'yanov, G.E. Stavroulakis, L.N. Polyakova and P.D. Panagiotopoulos: *Quasi-differentiability and Nonsmooth Modelling in Mechanics, Engineering and Economics.* 1996 ISBN 0-7923-4093-0
11. B. Mirkin: *Mathematical Classification and Clustering.* 1996 ISBN 0-7923-4159-7
12. B. Roy: *Multicriteria Methodology for Decision Aiding.* 1996 ISBN 0-7923-4166-X
13. R.B. Kearfott: *Rigorous Global Search: Continuous Problems.* 1996 ISBN 0-7923-4238-0
14. P. Kouvelis and G. Yu: *Robust Discrete Optimization and Its Applications.* 1997 ISBN 0-7923-4291-7
15. H. Konno, P.T. Thach and H. Tuy: *Optimization on Low Rank Nonconvex Structures.* 1997 ISBN 0-7923-4308-5
16. M. Hajdu: *Network Scheduling Techniques for Construction Project Management.* 1997 ISBN 0-7923-4309-3
17. J. Mockus, W. Eddy, A. Mockus, L. Mockus and G. Reklaitis: *Bayesian Heuristic Approach to Discrete and Global Optimization.* Algorithms, Visualization, Software, and Applications. 1997 ISBN 0-7923-4327-1
18. I.M. Bomze, T. Csendes, R. Horst and P.M. Pardalos (eds.): *Developments in Global Optimization.* 1997 ISBN 0-7923-4351-4
19. T. Rapcsák: Smooth Nonlinear Optimization in R^n. 1997 ISBN 0-7923-4680-7
20. A. Migdalas, P.M. Pardalos and P. VÎrbrand (eds.): *Multilevel Optimization: Algorithms and Applications.* 1998 ISBN 0-7923-4693-9
21. E.S. Mistakidis and G.E. Stavroulakis: *Nonconvex Optimization in Mechanics.* Algorithms, Heuristics and Engineering Applications by the F.E.M. 1998 ISBN 0-7923-4812-5

22. H. Tuy: *Convex Analysis and Global Optimization.* 1998 ISBN 0-7923-4818-4
23. D. Cieslik: *Steiner Minimal Trees.* 1998 ISBN 0-7923-4983-0
24. N.Z. Shor: *Nondifferentiable Optimization and Polynomial Problems.* 1998 ISBN 0-7923-4997-0
25. R. Reemtsen and J.-J. Rückmann (eds.): *Semi-Infinite Programming.* 1998 ISBN 0-7923-5054-5
26. B. Ricceri and S. Simons (eds.): *Minimax Theory and Applications.* 1998 ISBN 0-7923-5064-2
27. J.-P. Crouzeix, J.-E. Martinez-Legaz and M. Volle (eds.): *Generalized Convexitiy, Generalized Monotonicity: Recent Results.* 1998 ISBN 0-7923-5088-X
28. J. Outrata, M. Kočvara and J. Zowe: *Nonsmooth Approach to Optimization Problems with Equilibrium Constraints.* 1998 ISBN 0-7923-5170-3
29. D. Motreanu and P.D. Panagiotopoulos: *Minimax Theorems and Qualitative Properties of the Solutions of Hemivariational Inequalities.* 1999 ISBN 0-7923-5456-7
30. J.F. Bard: *Practical Bilevel Optimization.* Algorithms and Applications. 1999 ISBN 0-7923-5458-3
31. H.D. Sherali and W.P. Adams: *A Reformulation-Linearization Technique for Solving Discrete and Continuous Nonconvex Problems.* 1999 ISBN 0-7923-5487-7
32. F. Forgó, J. Szép and F. Szidarovszky: *Introduction to the Theory of Games.* Concepts, Methods, Applications. 1999 ISBN 0-7923-5775-2
33. C.A. Floudas and P.M. Pardalos (eds.): *Handbook of Test Problems in Local and Global Optimization.* 1999 ISBN 0-7923-5801-5
34. T. Stoilov and K. Stoilova: *Noniterative Coordination in Multilevel Systems.* 1999 ISBN 0-7923-5879-1
35. J. Haslinger, M. Miettinen and P.D. Panagiotopoulos (eds.): *Finite Element Method for Hemivariational Inequalities.* Theory, Methods and Applications. 1999 ISBN 0-7923-5951-8
36. V. Korotkich: *A Mathematical Structure of Emergent Computation.* 1999 ISBN 0-7923-6010-9
37. C.A. Floudas: *Deterministic Global Optimization: Theory, Methods and Applications.* 2000 ISBN 0-7923-6014-1
38. F. Giannessi (ed.): *Vector Variational Inequalities and Vector Equilibria.* Mathematical Theories. 1999 ISBN 0-7923-6026-5

KLUWER ACADEMIC PUBLISHERS – DORDRECHT / BOSTON / LONDON