Engineering Optimization

An Introduction with Metaheuristic Applications

Xin-She Yang
University of Cambridge
Department of Engineering
Cambridge, United Kingdom

WILEY
A JOHN WILEY & SONS, INC., PUBLICATION

Published by John Wiley & Sons, Inc., Hoboken, New Jersey.

Published simultaneously in Canada.

Library of Congress Cataloging-in-Publication Data:

Yang, Xin-She.
Engineering optimization : an introduction with metaheuristic applications / Xin-She Yang.
p. cm.
Includes bibliographical references and index.
ISBN 978-0-470-58246-6 (cloth)
1. Heuristic programming. 2. Mathematical optimization. 3. Engineering mathematics. I. Title.
T57.84.Y36 2010
620.001'5196—dc22 2010003429

10 9 8 7 6 5 4 3 2 1

CONTENTS

List of Figures xiii
Preface xix
Acknowledgments xxi
Introduction xxiii

PART I FOUNDATIONS OF OPTIMIZATION AND ALGORITHMS

1 A Brief History of Optimization 3

1.1 Before 1900 4
1.2 Twentieth Century 6
1.3 Heuristics and Metaheuristics 7
Exercises 10

2 Engineering Optimization 15

2.1 Optimization 15
2.2 Type of Optimization 17
2.3 Optimization Algorithms 19
2.4 Metaheuristics 22
2.5 Order Notation 22

2.6 Algorithm Complexity 24
2.7 No Free Lunch Theorems 25
Exercises 27

3 Mathematical Foundations 29

3.1 Upper and Lower Bounds 29
3.2 Basic Calculus 31
3.3 Optimality 35
3.3.1 Continuity and Smoothness 35
3.3.2 Stationary Points 36
3.3.3 Optimality Criteria 38
3.4 Vector and Matrix Norms 40
3.5 Eigenvalues and Definiteness 43
3.5.1 Eigenvalues 43
3.5.2 Definiteness 46
3.6 Linear and Affine Functions 48
3.6.1 Linear Functions 48
3.6.2 Affine Functions 49
3.6.3 Quadratic Form 49
3.7 Gradient and Hessian Matrices 51
3.7.1 Gradient 51
3.7.2 Hessian 51
3.7.3 Function approximations 52
3.7.4 Optimality of multivariate functions 52
3.8 Convexity 53
3.8.1 Convex Set 53
3.8.2 Convex Functions 55
Exercises 58

4 Classic Optimization Methods I 61

4.1 Unconstrained Optimization 61
4.2 Gradient-Based Methods 62
4.2.1 Newton's Method 62
4.2.2 Steepest Descent Method 63
4.2.3 Line Search 65
4.2.4 Conjugate Gradient Method 66
4.3 Constrained Optimization 68
4.4 Linear Programming 68

4.5 Simplex Method 70
4.5.1 Basic Procedure 70
4.5.2 Augmented Form 72
4.6 Nonlinear Optimization 76
4.7 Penalty Method 76
4.8 Lagrange Multipliers 76
4.9 Karush-Kuhn-Tucker Conditions 80
Exercises 83

5 Classic Optimization Methods II 85

5.1 BFGS Method 85
5.2 Nelder-Mead Method 86
5.2.1 A Simplex 86
5.2.2 Nelder-Mead Downhill Simplex 86
5.3 Trust-Region Method 88
5.4 Sequential Quadratic Programming 91
5.4.1 Quadratic Programming 91
5.4.2 Sequential Quadratic Programming 91
Exercises 93

6 Convex Optimization 95

6.1 KKT Conditions 95
6.2 Convex Optimization Examples 97
6.3 Equality Constrained Optimization 99
6.4 Barrier Functions 101
6.5 Interior-Point Methods 104
6.6 Stochastic and Robust Optimization 105
Exercises 107

7 Calculus of Variations 111

7.1 Euler-Lagrange Equation 111
7.1.1 Curvature 111
7.1.2 Euler-Lagrange Equation 114
7.2 Variations with Constraints 120
7.3 Variations for Multiple Variables 124
7.4 Optimal Control 125
7.4.1 Control Problem 126
7.4.2 Pontryagin's Principle 127

7.4.3 Multiple Controls 129
7.4.4 Stochastic Optimal Control 130
Exercises 131

8 Random Number Generators 133

8.1 Linear Congruential Algorithms 133
8.2 Uniform Distribution 134
8.3 Other Distributions 136
8.4 Metropolis Algorithms 140
Exercises 141

9 Monte Carlo Methods 143

9.1 Estimating π 143
9.2 Monte Carlo Integration 146
9.3 Importance of Sampling 149
Exercises 151

10 Random Walk and Markov Chain 153

10.1 Random Process 153
10.2 Random Walk 155
10.2.1 1D Random Walk 156
10.2.2 Random Walk in Higher Dimensions 158
10.3 Lévy Flights 159
10.4 Markov Chain 161
10.5 Markov Chain Monte Carlo 161
10.5.1 Metropolis-Hastings Algorithms 164
10.5.2 Random Walk 166
10.6 Markov Chain and Optimisation 167
Exercises 169

PART II METAHEURISTIC ALGORITHMS

11 Genetic Algorithms 173

11.1 Introduction 173
11.2 Genetic Algorithms 174
11.2.1 Basic Procedure 174
11.2.2 Choice of Parameters 176
11.3 Implementation 177

Exercises 179

12 Simulated Annealing 181

12.1 Annealing and Probability 181
12.2 Choice of Parameters 182
12.3 SA Algorithm 184
12.4 Implementation 184
Exercises 186

13 Ant Algorithms 189

13.1 Behaviour of Ants 189
13.2 Ant Colony Optimization 190
13.3 Double Bridge Problem 192
13.4 Virtual Ant Algorithm 193
Exercises 195

14 Bee Algorithms 197

14.1 Behavior of Honey Bees 197
14.2 Bee Algorithms 198
14.2.1 Honey Bee Algorithm 198
14.2.2 Virtual Bee Algorithm 200
14.2.3 Artificial Bee Colony Optimization 201
14.3 Applications 201
Exercises 202

15 Particle Swarm Optimization 203

15.1 Swarm Intelligence 203
15.2 PSO algorithms 204
15.3 Accelerated PSO 205
15.4 Implementation 207
15.4.1 Multimodal Functions 207
15.4.2 Validation 208
15.5 Constraints 209
Exercises 210

16 Harmony Search 213

16.1 Music-Based Algorithms 213

16.2 Harmony Search 215
16.3 Implementation 217
Exercises 218

17 Firefly Algorithm 221

17.1 Behaviour of Fireflies 221
17.2 Firefly-Inspired Algorithm 222
17.2.1 Firefly Algorithm 222
17.2.2 Light Intensity and Attractiveness 222
17.2.3 Scaling and Global Optima 225
17.2.4 Two Special Cases 225
17.3 Implementation 226
17.3.1 Multiple Global Optima 226
17.3.2 Multimodal Functions 227
17.3.3 FA Variants 228
Exercises 229

PART III APPLICATIONS

18 Multiobjective Optimization 233

18.1 Pareto Optimality 233
18.2 Weighted Sum Method 237
18.3 Utility Method 239
18.4 Metaheuristic Search 241
18.5 Other Algorithms 242
Exercises 244

19 Engineering Applications 247

19.1 Spring Design 247
19.2 Pressure Vessel 248
19.3 Shape Optimization 249
19.4 Optimization of Eigenvalues and Frequencies 252
19.5 Inverse Finite Element Analysis 256
Exercises 258

Appendices 261

Appendix A: Test Problems in Optimization 261

Appendix B: Matlab® Programs 267
B.1 Genetic Algorithms 267
B.2 Simulated Annealing 270
B.3 Particle Swarm Optimization 272
B.4 Harmony Search 273
B.5 Firefly Algorithm 275
B.6 Large Sparse Linear Systems 278
B.7 Nonlinear Optimization 279
B.7.1 Spring Design 279
B.7.2 Pressure Vessel 281

Appendix C: Glossary 283

Appendix D: Problem Solutions 305

References 333

Index 343

INTRODUCTION

Optimization can mean many different things. However, mathematically speaking, it is possible to write an optimization problem in the generic form

$$\underset{\boldsymbol{x}\in\Re^n}{\text{minimize}} \quad f_i(\boldsymbol{x}), \qquad (i = 1, 2, ..., M), \tag{I.1}$$

$$\text{subject to} \quad \phi_j(\boldsymbol{x}) = 0, \quad (j = 1, 2, ..., J), \tag{I.2}$$

$$\psi_k(\boldsymbol{x}) \leq 0, \quad (k = 1, 2, ..., K), \tag{I.3}$$

where $f_i(\boldsymbol{x}), \phi_j(\boldsymbol{x})$ and $\psi_k(\boldsymbol{x})$ are functions of the design vector

$$\boldsymbol{x} = (x_1, x_2, ..., x_n)^T, \tag{I.4}$$

where the components x_i of $\boldsymbol{x}$ are called design or decision variables, and they can be real continuous, discrete or a mixture of these two. The functions $f_i(\boldsymbol{x})$ where $i = 1, 2, ..., M$ are called the objective functions, and in the case of $M = 1$, there is only a single objective. The objective function is sometimes called the cost function or energy function in literature. The space spanned by the decision variables is called the search space $\Re^n$, while the space formed by the objective function values is called the solution space.

The objective functions can be either linear or nonlinear. The equalities for ϕ_j and inequalities for ψ_k are called constraints. It is worth pointing out

that we can also write the inequalities in the other way ≥ 0, and we can also formulate the objectives as a maximization problem. This is because the maximization of $f(\boldsymbol{x})$ is equivalent to the minimization of $-f(\boldsymbol{x})$, and any inequality $g(\boldsymbol{x}) \leq 0$ is equivalent to $-g(\boldsymbol{x}) \geq 0$. For the constraints, the simplest case for a decision variable x_i is $x_{i,\min} \leq x_i \leq x_{i,\max}$, which is called bounds.

If the constraints ϕ_j and ψ_k are all linear, then it becomes a linearly constrained problem. If both the constraints and the objective functions are all linear, it becomes a linear programming problem. For linear programming problems, a significant progress was the development of the simplex method in 1947 by George B. Dantzig. However, generally speaking, since all f_i, ϕ_j and ψ_k are nonlinear, we have to deal with a nonlinear optimization problem. It is worth pointing out that all the functions (objective and constraints) are collectively called problem functions.

A special class of optimization is when there is no constraint at all (or $J = K = 0$), and the only task is to find the minimum or maximum of a single objective function $f(\boldsymbol{x})$. This usually makes things much easier, though not always. In this case, the optimization problem becomes an unconstrained one.

For example, we can find the minimum of the Rosenbrock banana function

$$f(x, y) = (1 - x)^2 + 100(y - x^2)^2. \tag{I.5}$$

In order to find its minimum, we can set its partial derivatives to zero, and we have

$$\frac{\partial f}{\partial x} = 2(1 - x) - 400(y - x^2)x = 0, \tag{I.6}$$

$$\frac{\partial f}{\partial y} = 200(y - x^2) = 0. \tag{I.7}$$

The second equation implies that $y = x^2$ can be substituted into the first one. We have

$$1 - x - 200(x^2 - x^2) = 1 - x = 0, \tag{I.8}$$

or $x = 1$. The minimum $f_{\min} = 0$ occurs at $x = y = 1$. This method uses important information from the objective function; that is, the gradient or first derivatives. Consequently, we can use gradient-based optimization methods such as Newton's method and conjugate gradient methods to find the minimum of this function.

A potential problem arises when we do not know the the gradient, or the first derivatives do not exist or are not defined. For example, we can design the following function

$$f(x, y) = (|x| + |y|) \exp[-\sin(x^2) - \sin(y^2)]. \tag{I.9}$$

The global minimum occurs at $(x, y) = (0, 0)$, but the derivatives at $(0, 0)$ are not well defined due to the factor $|x| + |y|$ and there is some discontinuity in the first derivatives. In this case, it is not possible to use gradient-based

optimization methods. Obviously, we can use gradient-free method such as the Nelder-Mead downhill simplex method. But as the objective function is multimodal (because of the sine function), such optimization methods are very sensitive to the starting point. If the starting point is far from the the sought minimum, the algorithm will usually get stuck in a local minimum and/or simply fail.

Optimization can take other forms as well. Many mathematical and statistical methods are essentially a different form of optimization. For example, in data processing, the methods of least squares try to minimize the sum of the residuals or differences between the predicated values (by mathematical models) and the observed values. All major numerical methods such as finite difference methods intend to find some approximations that minimize the difference of the true solutions and the estimated solutions. In aircraft design, we try to design the shape in such a way so as to minimize the drag and maximize the lifting force. All these formulations could be converted or related to the generic form of the nonlinear optimization formulation discussed above. In some extreme cases, the objective functions do not have explicit form, or at least it cannot be easily linked with the design variables. For example, nowadays in product design and city planning, we have to optimize the energy efficiency and minimize the environmental impact. The study of such impact itself is a challenging topic and it is not always easy to characterize them; however, we still try to find some suboptimal or even optimal solutions in this context.

Nonlinearity and multimodality are the main problem, which renders most conventional methods such as the hill-climbing method inefficient and stuck in the wrong solutions. Another even more challenging problem arises when the number of decision variables increases or n is very large, say, $n = 50,000$. In addition, the nonlinearity coupled with the large scale complexity makes things even worse. For example, the well-known traveling salesman problem is to try to find the shortest route for a salesman to travel n cities once and only once. The number of possible combinations, without knowing the distribution of the cities, is $n!$. If $n = 100$, this number of combinations $n! \approx 9.3 \times 10^{157}$ is astronomical. The top supercomputers in the world such as IBM's Blue Gene can now do about 3 petraflops; there are about 3×10^{15} floating-point operations per second. In fact, with all the available computers in the world fully dedicated to the brutal force search of all the combinations of 100!, it would take much longer than the lifetime of the known universe. This clearly means that it is not practical to search all possible combinations. We have to use some alternative, yet efficient enough, methods.

Heuristic and metaheuristic algorithms are designed to deal with this type of problem. Most these algorithms are nature-inspired or bio-inspired as they have been developed based on the successful evolutionary behavior of natural systems – by learning from nature. Nature has been solving various tough problems over millions or even billions of years. Only the best and robust solutions remain – survival of the fittest. Similarly, heuristic algorithms use

the trial-and-error, learning and adaptation to solve problems. We cannot expect them to find the best solution all the time, but expect them to find the good enough solutions or even the optimal solution most of the time, and more importantly, in a reasonably and practically short time. Modern metaheuristic algorithms are almost guaranteed to work well for a wide range of tough optimization problems. However, it is a well-known fact that there is 'no free lunch' in optimization. It has been proved by Wolpert and Macready in 1997 that if algorithm A is better than algorithm B for some problems, then B will outperform A for other problems. That is to say, a universally efficient algorithm does not exist. The main aim of research in optimization and algorithm development is to design and/or choose the most suitable and efficient algorithms for a given optimization task.

Loosely speaking, modern metaheuristic algorithms for engineering optimization include genetic algorithms (GA), simulated annealing (SA), particle swarm optimization (PSO), ant colony algorithm, bee algorithm, harmony search (HS), firefly algorithm (FA), and many others.

We will introduce all the major and widely used metaheuristics in Part II of this book, after a detailed introduction to the fundamentals of engineering optimization. In Part III, we will briefly outline other important algorithms and multiobjective optimization. We then focus on the applications of the algorithms introduced in the book to solve real-world optimization problems.

Each chapter will be self-contained or with minimal cross references to other chapters. We will include some exercises at the end of each chapter with detailed answers in the appendices. A further reading list is also provided at the end of each chapter. These make it ideal for the book to be used either as a textbook for relevant courses, or an additional reference as well as for self study. The self-contained nature of each chapter means that lecturers and students can use each individual chapter to suit their own purpose.

The main requirement for this book is the basic understanding of the calculus, particularly differentiation, and a good undestanding of algebraic manipulations. We will try to review these briefly in Part I.

In addition, the implementation of algorithms will inevitably use a programming language. However, we believe that the efficiency and performance of each algorithm, if properly implemented, should be independent of any programming. For this reason, we will explain each algorithm as detail as possible, but leave an actual implementation in the appendices where we will include some simple Matlab/Octave programs for demonstrating how the implemented algorithms work.

REFERENCES

1. G. B. Dantzig, *Linear Programming and Extensions*, Princeton University Press, 1963.

2. P. E. Gill, W. Murray, and M. H. Wright, *Practical Optimization*, Academic Press Inc., 1981.

3. S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing", *Science*, **220** (4598), 671-680 (1983).

4. D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimizaiton", *IEEE Transaction on Evolutionary Computation*, **1**, 67-82 (1997).

5. X. S. Yang, "Harmony search as a metaheuristic algorithm", in: *Music-Inspired Harmony Search Algorithm: Theory and Applications* (eds. Z. W. Geem), Springer, p. 1-14 (2009).

CHAPTER 1

A BRIEF HISTORY OF OPTIMIZATION

Optimization is everywhere, from engineering design to financial markets, from our daily activity to planning our holidays, and computer sciences to industrial applications. We always intend to maximize or minimize something. An organization wants to maximize its profits, minimize costs, and maximize performance. Even when we plan our holidays, we want to maximize our enjoyment with least cost (or ideally free). In fact, we are constantly searching for the optimal solutions to every problem we meet, though we are not necessarily able to find such solutions.

It is no exaggeration to say that finding the solution to optimization problems, whether intentionally or subconsciously, is as old as human history itself. For example, the least effort principle can often explain many human behaviors. We know the shortest distance between any two different points on a plane is a straight line, though it often needs complex maths such as the calculus of variations to formally prove that a straight line segment between the two points is indeed the shortest.

In fact, many physical phenomena are governed by the so-called least action principle or its variants. For example, light travels and obeys Fermat's principle, that is to travel at the shortest time from one medium to another,

Engineering Optimization: An Introduction with Metaheuristic Applications.
By Xin-She Yang

thus resulting in Snell's law. The whole analytical mechanics is based on this least action principle.

1.1 BEFORE 1900

The study of optimization problems is also as old as science itself. It is known that the ancient Greek mathematicians solved many optimization problems. For example, Euclid in around 300BC proved that a square encloses the greatest area among all possible rectangles with the same total length of four sides. Later, Heron in around 100BC suggested that the distance between two points along the path reflected by a mirror is the shortest when light travels and reflects from a mirror obeying some symmetry, that is the angle of incidence is equal to the angle of reflection. It is a well-know optimization problem, called Heron's problem, as it was first described in Heron's *Catoptrica* (or *On Mirrors*).

The celebrated German astronomer, Johannes Kepler, is mainly famous for the discovery of his three laws of planetary motion; however, in 1613, he solved an optimal solution to the so-called marriage problem or secretary problem when he started to look for his second wife. He described his method in his personal letter dated October 23, 1613 to Baron Strahlendorf, including the balance of virtues and drawbacks of each candidate, her dowry, hesitation, and advice of friends. Among the eleven candidates interviewed, Kepler chose the fifth, though his friend suggested him to choose the fourth candidate. This may imply that Kepler was trying to optimize some utility function of some sort. This problem was formally introduced by Martin Gardner in 1960 in his *mathematical games* column in the February 1960 issue of *Scientific American*. Since then, it has developed into a field of probability optimization such as optimal stopping problems.

W. van Royen Snell discovered in 1621 the law of refraction, which remained unpublished; later, Christiaan Huygens mentioned Snell's results in his *Dioptrica* in 1703. This law was independently rediscovered by René Descartes and published in his treatise *Discours de la Methode* in 1637. About 20 years later, when Descartes' students contacted Pierre de Fermat collecting his correspondence with Descartes, Fermat looked again in 1657 at his argument with the unsatisfactory description of light refraction by Descartes, and derived Snell and Descartes' results from a more fundamental principle – light always travels in the shortest time in any medium, and this principle for light is now referred to as *Fermat's principle*, which laid the foundation of modern optics.

In his *Principia Mathematica* published in 1687, Sir Isaac Newton solved the problem of the body shape of minimal resistance that he posed earlier in 1685 as a pioneering problem in optimization, now a problem of the calculus of variations. The main aim was to find the shape of a symmetrical revolution body so as to minimize the resistance to motion in a fluid. Subsequently,

Newton derived the resistance law of the body. Interestingly, Galileo Galilei independently suggested a similar problem in 1638 in his *Discursi.*

In June 1696, J. Bernoulli made some significant progress in calculus. In an article in *Acta Eruditorum*, he challenged all the mathematicians in the world to find the shape or curve connecting two points at different heights so that a body will fall along the curve in the shortest time due to gravity – the line of quickest descent, though Bernoulli already knew the solution. On January 29, 1697 the challenge was received by Newton when he come home at four in the afternoon and he did not sleep until he had solved it by about four the next morning and on the same day he sent out his solution. Though Newton managed to solve it in less than 12 hours as he became the Warden of the Royal Mint on March 19, 1696, some suggested that he, as such a genius, should have been able to solve it in half an hour. Some said this was the first hint or evidence that too much administrative work will slow down one's progress. The solution as we now know is a part of a cycloid. This steepest descent is now called Brachistochrone problem, which inspired Euler and Lagrange to formulate the general theory of calculus of variations.

In 1746, the *principle of least action* was proposed by P. L. de Maupertuis to unify various laws of physical motion and its application to explain all phenomena. In modern terminology, it is a variational principle of stationary action in terms of an integral equation of a functional in the framework of calculus of variations, which plays a central role in the Lagrangian and Hamiltonian classical mechanics. It is also an important principle in mathematics and physics.

In 1781, Gaspard Monge, a French civil engineer, investigated the transportation problem for optimal transportation and allocation of resources, if the initial and final spatial distribution are known. In 1942, Leonid Kantorovich showed that this combinatorial optimization problem is in fact a case of a linear programming problem.

Around 1801, Frederich Gauss claimed that he used the *method of least-squares* to predict the orbital location of the asteroid Ceres, though his version of the least squares with more rigorous mathematical foundation was published later in 1809. In 1805, Adrien Legendre was the first to describe the method of least squares in an appendix of his book *Nouvelle meéthodes pour la determination des orbites des cometes*, and in 1806 he used the principle of least squares for curve fitting. Gauss later claimed that he had been using this method for more than 20 years, and laid the foundation for least-squares analysis in 1795. This led to some bitter disputes with Legendre. In 1808, Robert Adrain, unaware of Legendre's work, published the method of least squares studying the uncertainty and errors in making observations, not using the same terminology as those by Legendre.

In 1815, D. Ricardo proposed the *law of diminishing returns* for land cultivation, which can be applied in many activities. For example, the productivity of a piece of a land or a factory will only increase marginally with additional increase of inputs. This law is called law of increasing opportunity cost. It

dictates that there is a fundamental relationship between opportunity and scarcity of resources, thus requiring that scarcely available resources be used efficiently.

In 1847 in a short note, L. A. Cauchy proposed a general method for solving systems of equations in an iterative way. This essentially leads to two iterative methods of minimization: now called the gradient method and steepest descent, for certain functions of more than one variable.

1.2 TWENTIETH CENTURY

In 1906, Danish mathematician J. Jensen introduced the concept of convexity and derived an inequality, now referred to as Jensen's inequality, which plays an important role in convex optimization and other areas such as economics. Convex optimization is a special but very important class of mathematical optimization as any optimality found is also guaranteed to be the global optimality. A wider range of optimization problems can be reformulated in terms of convex optimization. Consequently, it has many applications including control systems, data fitting and modelling, optimal design, signal processing, mathematical finance, and others.

As early as 1766, Leonhard Euler studied the Knight tour problem, and T. P. Kirkman published a research article on the way to find a circuit which passes through each vertex once and only once for a give graph of polyhedra. In 1856, Sir William Rowan Hamilton popularized his *Icosian Game*. Then, in February 1930, Karl Menger posed the Messenger's problem at a mathematical colloquium in Vienna, as this problem is often encountered by postal messengers and travelers. His work was published later in 1932. The task is to find the shortest path connecting a finite number of points/cities whose pairwise distances are known. Though the problem is solvable in a finite number of trials and permutations, there is no efficient algorithm for finding such solutions. In general, the simple rule of going to the nearest points does not result in the shortest path. This problem is now referred to as *Traveling Salesman Problem* which is closely related to many different applications such as network routing, resource allocation, scheduling and operations research in general. In fact, as early as 1832, the 1832 traveling salesman manual described a tour along 45 German cities with a shortest route of 1248 km, though the exact mathematical roots of this problem are quite obscure, and might be well around for some time before the 1930s.

Interestingly, H. Hancock published in 1917 the first book on optimization "*Theory of Minima and Maxima*".

In 1939, L. Kantorovich was the first to develop an algorithm for linear programming and use it in economics. He formulated the production problem of optimal planning and effective methods for finding solutions using linear programming. For this work, he shared the Noble prize with T. Koopmans in 1975. The next important step of progress is that George Dantzig invented in

1947 the *simplex method* for solving large-scale linear programming problems. Dorfman in an article published in 1984 wrote that linear programming was discovered three times, independently, between 1939 and 1947, but each time in a somewhat different form. The first discovery was by the Russian mathematician, L. Kantorovich, then by the Dutch economist, Koopmans, and the third in 1947 by the American mathematician George Dantzig. Dantzig's revolutionary simplex method is able to solve a wide range of optimal policy decision problems of great complexity. A classic example and one of the earliest of using linear programming as described in Dantzig's 1963 book was to find the solution to the special optimal diet problem involving 9 equations and 77 unknowns using hand-operated desk calculators.

In 1951, Harold Kuhn and A. W. Tucker studied the nonlinear optimiation problem and re-developed the optimality condition, as similar conditions were proposed by W. Karush in 1939 in his MSc dissertation. In fact, the optimality conditions are the generalization of Lagrange multipliers to nonlinear inequalities, and are now known as the Karush-Kuhn-Tucker conditions, or simply Kuhn-Tucker conditions, which are necessary conditions for a solution to be optimal in nonlinear programming.

Then, in 1957, Richard Bellman at Stanford University developed the dynamic programming and the optimality principle when studying multistage decision and planning processes while he spent some time at the RAND Corporation. He also coined the term *Dynamic Programming*. The idea of dynamic programming can date back to 1944 when John von Neumann and O. Morgenstern studied the sequential decision problems. John von Neumann also made important contribution to the development of operational research. As earlier as in 1840, Charles Babbage studied the cost of transportation and sorting mails; this could be the earliest research on the operational research. Significant progress was made during the Second World War, and ever since it expanded to find optimal or near optimal solutions in a wide range complex problems of interdisciplinary areas such as communication networks, project planning, scheduling, transport planning, and management.

After the 1960s, the literature on optimization exploded, and it would take a whole book to write even a brief history on optimization after the 1960s. As this book is mainly about the introduction to metaheuristic algorithms, we will then focus our attention on the development of heuristics and metaheursitics. In fact, quite a significant number of new algorithms in optimization are primarily metaheuristics.

1.3 HEURISTICS AND METAHEURISTICS

Heuristics is a solution strategy by trial-and-error to produce acceptable solutions to a complex problem in a reasonably practical time. The complexity of the problem of interest makes it impossible to search every possible solution or combination, the aim is to find good, feasible solutions in an acceptable

timescale. There is no guarantee that the best solutions can be found, and we even do not know whether an algorithm will work and why if it does work. The idea is that an efficient but practical algorithm that will work most of the time and be able to produce good quality solutions. Among the found quality solutions, it is expected that some of them are nearly optimal, though there is no guarantee for such optimality.

Alan Turing was probably the first to use heuristic algorithms during the Second World War when he was breaking German Enigma ciphers at Bletchley Park where Turing, together with British mathematician Gordon Welchman, designed in 1940 a cryptanalytic electromechanical machine, the *Bombe*, to aid their code-breaking work. The bombe used a heuristic algorithm, as Turing called, to search, among about 10^{22} potential combinations, the possibly correct setting coded in an Enigma message. Turing called his search method *heuristic search*, as it could be expected it worked most of the time, but there was no guarantee to find the correct solution, but it was a tremendous success. In 1945, Turing was recruited to the National Physical Laboratory (NPL), UK where he set out his design for the Automatic Computing Engine (ACE). In an NPL report on "Intelligent machinery' in 1948, he outlined his innovative ideas of machine intelligence and learning, neural networks and evolutionary algorithms or an early version of genetic algorithms.

The next significant step is the development of evolutionary algorithms in the 1960s and 1970s. First, John Holland and his collaborators at the University of Michigan developed the genetic algorithms in the 1960s and 1970s. As early as 1962, Holland studied the adaptive system and was the first to use crossover and recombination manipulations for modeling such systems. His seminal book summarizing the development of genetic algorithms was published in 1975. In the same year, Kenneth De Jong finished his important dissertation showing the potential and power of genetic algorithms for a wide range of objective functions, either noisy, multimodal or even discontinuous.

Genetic algorithms (GA) is a search method based on the abstraction of Darwin's evolution and natural selection of biological systems and representing them in the mathematical operators: crossover or recombination, mutation, fitness, and selection of the fittest. Ever since, genetic algorithms become so successful in solving a wide range of optimization problems, several thousand research articles and hundreds of books have been written. Some statistics show that a vast majority of Fortune 500 companies are now using them routinely to solve tough combinatorial optimization problems such as planning, data-fitting, and scheduling.

During the same period, Ingo Rechenberg and Hans-Paul Schwefel both then at the Technical University of Berlin developed a search technique for solving optimization problem in aerospace engineering, called evolution strategy, in 1963. Later, Peter Bienert joined them and began to construct an automatic experimenter using simple rules of mutation and selection. There is no crossover in this technique; only mutation was used to produce an offspring and an improved solution was kept at each generation. This is essentially a

simple trajectory-style hill-climbing algorithm with randomization. As early as 1960, Lawrence J. Fogel intended to use simulated evolution as a learning process as a tool to study artificial intelligence. Then, in 1966, L. J. Fogel, with A. J. Owen and M. J. Walsh, developed the evolutionary programming technique by representing solutions as finite-state machines and randomly mutating one of these machines. The above innovative ideas and methods have evolved into a much wider discipline, called evolutionary algorithms and evolutionary computation.

The decades of the 1980s and 1990s were the most exciting time for metaheuristic algorithms. The next big step is the development of simulated annealing (SA) in 1983, an optimization technique, pioneered by S. Kirkpatrick, C. D. Gellat and M. P. Vecchi, inspired by the annealing process of metals. It is a trajectory-based search algorithm starting with an initial guess solution at a high temperature, and gradually cooling down the system. A move or new solution is accepted if it is better; otherwise, it is accepted with a probability, which makes it possible for the system to escape any local optima. It is then expected that if the system is cooled slowly enough, the global optimal solution can be reached.

The actual first usage of metaheuristic is probably due to Fred Glover's Tabu search in 1986, though his seminal book on Tabu search was published later in 1997.

In 1992, Marco Dorigo finished his PhD thesis on optimization and natural algorithms, in which he described his innovative work on ant colony optimization (ACO). This search technique was inspired by the swarm intelligence of social ants using pheromone as a chemical messenger. Then, in 1992, John R. Koza of Stanford University published a treatise on genetic programming which laid the foundation of a whole new area of machine learning, revolutionizing computer programming. As early as in 1988, Koza applied his first patent on genetic programming. The basic idea is to use the genetic principle to breed computer programs so as to produce the best programs for a given type of problem.

Slightly later in 1995, another significant step of progress is the development of the particle swarm optimization (PSO) by American social psychologist James Kennedy, and engineer Russell C. Eberhart. Loosely speaking, PSO is an optimization algorithm inspired by the swarm intelligence of fish and birds and even by human behavior. The multiple agents, called particles, swarm around the search space starting from some initial random guess. The swarm communicates the current best and shares the global best so as to focus on the quality solutions. Since its development, there have been about 20 different variants of particle swarm, and have been applied to almost all areas of tough optimization problems. There is some strong evidence that PSO is better than traditional search algorithms and even better than genetic algorithms for most type of problems, though this is far from conclusive.

In 1997, the publication of the 'no free lunch theorems for optimization' by D. H. Wolpert and W. G. Macready sent out a shock wave to the optimization

community. Researchers have always been trying to find better algorithms, or even universally robust algorithms, for optimization, especially for tough NP-hard optimization problems. However, these theorems state that if algorithm A performs better than algorithm B for some optimization functions, then B will outperform A for other functions. That is to say, if averaged over all possible function space, both algorithms A and B will perform on average equally well. Alternatively, there is no universally better algorithms exist. That is disappointing, right? Then, people realized that we do not need the average over all possible functions as for a given optimization problem. What we want is to find the best solutions; this has nothing to do with average over all the whole function space. In addition, we can accept the fact that there is no universal or magical tool, but we do know from our experience that some algorithms indeed outperform others for given types of optimization problems. So the research now focuses on finding the best and most efficient algorithm(s) for a given problem. The task is to design better algorithms for most types of problems, not for all the problems. Therefore, the search is still on.

At the turn of the twenty-first century, things became even more exciting. First, Zong Woo Geem *et al.* in 2001 developed the Harmony Search (HS) algorithm, which has been widely applied in solving various optimization problems such as water distribution, transport modelling and scheduling. In 2004, S. Nakrani and C. Tovey proposed the Honey Bee algorithm and its application for optimizing Internet hosting centers, which followed by the development of a novel bee algorithm by D. T. Pham *et al.* in 2005 and the Artificial Bee Colony (ABC) by D. Karaboga in 2005. In 2008, the author of this book developed the Firefly Algorithm (FA). Quite a few research articles on the Firefly Algorithm then followed, and this algorithm has attracted a wide range of interests.

As we can see, more and more metaheuristic algorithms are being developed. Such a diverse range of algorithms necessitates a system summary of various metaheuristic algorithms, and this book is such an attempt to introduce all the latest and major metaheuristics with applications.

EXERCISES

1.1 Find the minimum value of $f(x) = x^2 - x - 6$ in $[-\infty, \infty]$.

1.2 For the previous problem, use simple differentiation to obtain the same results.

1.3 In about 300BC, Euclid proved that a square encloses the greatest area among all rectangles, assuming the total length of the four edges is fixed. Provide your own version of such proof.

1.4 Design a cylindrical water tank which uses the minimal materials and holds the largest volume of water. What is the relationship between the radius r of the base and the height h?

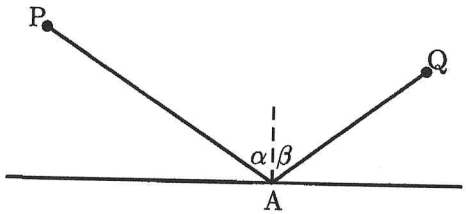

Figure 1.1: Reflection of light at a mirror.

1.5 In about 100BC, Heron proved that a path PAQ is the shortest when reflecting at a mirror with the angle of incidence α is equal to angle of reflectance β (see Figure 1.1). Show that $\alpha = \beta$ leads to the shortest distance PAQ.

REFERENCES

1. M. S. Bazaraa, H. D. Serali, and C. M. Shetty, *Nonlinear Programming: Theory and Algorithms*, John Wiley & Sons, 1993.
2. L. A. Cauchy, "Méthode générale pour la résolution des systeèmes d'équations simulatanées, *Comptes Rendus de l'Académie de Sciences de Paris*, **25**, 536-538 (1847).
3. B. J. Copeland, *The Essential Turing*, Oxford University Press, 2004.
4. B. J. Copeland, *Alan Turing's Automatic Computing Engine*, Oxford University Press, 2005.
5. K. De Jong, *Analysis of the Behaviour of a Class of Genetic Adaptive Systems*, PhD thesis, University of Michigan, Ann Anbor, 1975.
6. R Dorfman, "The discovery of linear programming", *Ann. Hist. Comput.*, **6** (3), 283-295 (1984).
7. M. Dorigo, *Optimization, Learning and Natural Algorithms*, PhD thesis, Politecnico di Milano, Italy, 1992.
8. S. Dreyfus, "Richard Bellman on the birth of dynamic programming", *Operations Research*, **50** (1), 48-51 (2002).
9. T. S. Ferguson, "Who solved the secretary problem?", *Statist. Sci.*, **4** (3), 282-296 (1989).
10. R. P. Feynman, R. B. Leighton, and M. Sands, *The Feynman Lectures on Physics*, vol. 2, Addison-Wesley, Reading, Mass., 1963.
11. L. J. Fogel, A. J. Owens, and M. J. Walsh, *Artificial Intelligence Through Simulated Evolution*, John Wiley & Sons, 1966.
12. Z. W. Geem, J. H. Kim and G. V. Loganathan, "A new heuristic optimization: Harmony search", *Simulation*, **76**(2), 60-68 (2001).
13. F. Glover, "Future paths for integer programming and links to artificial intelligence", *Comput. Operational Res.*, **13** (5), 533-549 (1986).

14. F. Glover and M. Laguna, *Tabu Search*, Kluwer Academic Publishers, Boston, 1997.

15. H. H. Holstine, *A History of the Calculus of Variations from the 17th through the 19th Century*, Springer-Verlag, Heidelberg, 1980.

16. A. A. Goldstein, "Cauchy's method of minimization", *Numerische Mathematik*, **4**(1), 146-150 (1962).

17. J. Holland, *Adaptation in Natural and Artficial systems*, University of Michigan Press, Ann Anbor, 1975.

18. J. L. W. V. Jensen, "Sur les fonctions convexes et les inégalités entre les valeurs moyenners", *Acta Mathematica*, **30**, 175-193 (1906).

19. P. Judea, *Heuristics*, Addison-Wesley, 1984.

20. D. Karaboga, "An idea based on honey bee swarm for numerical optimization", Technical Report, Erciyes University, 2005.

21. W. Karush, *Minima of Functions of Several Variables with Inequalities as Side Constraints*, MSc Dissertation, Department of Mathematics, University of Chicago, Illinois, 1939.

22. J. Kennedy and R. Eberhart, "Particle swarm optimization", in: *Proc. of the IEEE Int. Conf. on Neural Networks*, Piscataway, NJ, p. 1942-1948 (1995).

23. S. Kirkpatrick, C. D. Gellat, and M. P. Vecchi, "Optimization by simulated annealing", *Science*, **220**, 671-680 (1983).

24. J. R. Koza, *Genetic Programming: One the Programming of Computers by Means of Natural Selection*, MIT Press, 1992.

25. H. W. Kuhn and A. W. Tucker, "Nonlinear programming", *Proc. 2nd Berkeley Symposium*, University of California Press, Berkeley, p. 481-492 (1951).

26. K. Menger, "Dass botenproblem", in: *Ergebnisse eines Mathematischen Kolloquiums* **2**, Ed. K. Menger), Teubner, Leipzig, p. 11-12 (1932).

27. L. T. Moore, *Isaac Newton: A Biography*, Dover, New York, 1934.

28. S. Nakrani and C. Tovey, "On honey bees and dynamic server allocation in Internet hostubg centers", *Adaptive Behavior*, **12**, 223-240 (2004).

29. D. T. Pham, A. Ghanbarzadeh, E. Koc, S. Otri, S. Rahim and M. Zaidi, "The bees algorithm", Technical Note, Manufacturing Engineering Center, Cardiff University, 2005.

30. O. Sheynin, "On the history of the principle of least squares", *Archive for History of Exact Sciences*, **46** (1), 39-54 (1993).

31. A. Schrijver, "On the history of combinatorial optimization (till 1960)", in: *Handbook of Discrete Optimization* (Eds K. Aardal, G. L. Nemhauser, R. Weismantel), Elsevier, Amsterdam, p.1-68 (2005).

32. V. M. Tikhomirov, *Stories about Mixima and Minima*, American Mathematical Society, 1990.

33. A. M. Turing, *Intelligent Machinery*, National Physical Laboratory, 1948.

34. W. T. Tutte, *Graph Theory as I Have Known It*, Oxford Science Publications, 1998.

35. D. Wells, *The Penguin Book of Curious and Interesting Geometry*, Penguin, 1991.

36. D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization", *IEEE Transaction on Evolutionary Computation*, **1**, 67-82 (1997).

37. X. S. Yang, *Nature-Inspired Metaheuristic Algorithms*, Luniver Press, (2008).

38. X. S. Yang, "Firefly algorithms for multimodal optimization", *Proc. 5th Symposium on Stochastic Algorithms, Foundations and Applications, SAGA 2009*, Eds. O. Watanabe and T. Zeugmann, Lecture Notes in Computer Science, **5792**, 169-178 Japan, 2009.

39. History of optimization, http://hse-econ.fi/kitti/opthist.html

40. The Turing Archive for the History of Computing, http://www.alanturing.net/

41. History of Mathematics and Mathematicians, http://turnbull.dcs.st-and.ac.uk /history/Mathematicians

APPENDIX A

TEST PROBLEMS IN OPTIMIZATION

In order to validate any new optimization algorithm, we have to validate it against standard test functions so as to compare its performance with well-established or existing algorithms. There are many test functions, so there is no standard list or set of test functions one has to follow. However, various test functions do exist, so new algorithms should be tested using at least a subset of functions with diverse properties so as to make sure whether or not the tested algorithm can solve certain type of optimization efficiently.

In this appendix, we will provide a subset of commonly used test functions with simple bounds as constraints, though they are often listed as unconstrained problems in literature. We will list the function form $f(\boldsymbol{x})$, its search domain, optimal solutions $\boldsymbol{x}_*$ and/or optimal objective value f_*. Here, we use $\boldsymbol{x} = (x_1, ..., x_n)^T$ where n is the dimension.

Ackley's function:

$$f(\boldsymbol{x}) = -20 \exp\Big[-\frac{1}{5}\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\Big] - \exp\Big[\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\Big] + 20 + e, \quad \text{(A.1)}$$

Engineering Optimization: An Introduction with Metaheuristic Applications.
By Xin-She Yang

where $n = 1, 2, ...$, and $-32.768 \le x_i \le 32.768$ for $i = 1, 2, ..., n$. This function has the global minimum $f_* = 0$ at $\boldsymbol{x}_* = (0, 0, ..., 0)$.

De Jong's functions: The simplest of De Jong's functions is the so-called sphere function

$$f(\boldsymbol{x}) = \sum_{i=1}^{n} x_i^2, \qquad -5.12 \le x_i \le 5.12, \tag{A.2}$$

whose global minimum is obviously $f_* = 0$ at $(0, 0, ..., 0)$. This function is unimodal and convex. A related function is the so-called weighted sphere function or hyper-ellipsoid function

$$f(\boldsymbol{x}) = \sum_{i=1}^{n} i x_i^2, \qquad -5.12 \le x_i \le 5.12, \tag{A.3}$$

which is also convex and unimodal with a global minimum $f_* = 0$ at $\boldsymbol{x}_* = (0, 0, ..., 0)$. Another related test function is the sum of different power function

$$f(\boldsymbol{x}) = \sum_{i=1}^{n} |x_i|^{i+1}, \qquad -1 \le x_i \le 1, \tag{A.4}$$

which has a global minimum $f_* = 0$ at $(0, 0, ..., 0)$.

Easom's function:

$$f(\boldsymbol{x}) = -\cos(x)\cos(y)\exp\Big[-(x-\pi)^2 + (y-\pi)^2\Big], \tag{A.5}$$

whose global minimum is $f_* = -1$ at $\boldsymbol{x}_* = (\pi, \pi)$ within $-100 \le x, y \le 100$. It has many local minima. Xin-She Yang extended in 2008 this function to n dimensions, and we have

$$f(\boldsymbol{x}) = -(-1)^n \Big(\prod_{i=1}^{n} \cos^2(x_i)\Big) \exp\Big[-\sum_{i=1}^{n} (x_i - \pi)^2\Big], \tag{A.6}$$

whose global minimum $f_* = -1$ occurs at $\boldsymbol{x}_* = (\pi, \pi, ..., \pi)$. Here the domain is $-2\pi \le x_i \le 2\pi$ where $i = 1, 2, ..., n$.

Equality-Constrained Function:

$$f(\boldsymbol{x}) = -(\sqrt{n})^n \prod_{i=1}^{n} x_i, \tag{A.7}$$

subject to an equality constraint (a hyper-sphere)

$$\sum_{i=1}^{n} x_i^2 = 1. \tag{A.8}$$

The global minimum $f_* = -1$ of $f(\boldsymbol{x})$ occurs at $\boldsymbol{x}_*(1/\sqrt{n}, ..., 1/\sqrt{n})$ within the domain $0 \le x_i \le 1$ for $i = 1, 2, ..., n$.

Griewank's function:

$$f(\boldsymbol{x}) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos(\frac{x_i}{\sqrt{i}}) + 1, \qquad -600 \le x_i \le 600, \tag{A.9}$$

whose global minimum is $f_* = 0$ at $\boldsymbol{x}_* = (0, 0, ..., 0)$. This function is highly multimodal.

Michaelwicz's function:

$$f(\boldsymbol{x}) = -\sum_{i=1}^{n} \sin(x_i) \cdot \left[\sin(\frac{ix_i^2}{\pi})\right]^{2m}, \tag{A.10}$$

where $m = 10$, and $0 \le x_i \le \pi$ for $i = 1, 2, ..., n$. In 2D case, we have

$$f(x, y) = -\sin(x)\sin^{20}(\frac{x^2}{\pi}) - \sin(y)\sin^{20}(\frac{2y^2}{\pi}), \tag{A.11}$$

where $(x, y) \in [0, 5] \times [0, 5]$. This function has a global minimum $f_* \approx -1.8013$ at $\boldsymbol{x}_* = (x_*, y_*) = (2.20319, 1.57049)$.

Perm Functions:

$$f(\boldsymbol{x}) = \sum_{j=1}^{n} \Big\{ \sum_{i=1}^{n} (i^j + \beta)\Big[(\frac{x_i}{i})^j - 1\Big]\Big\}, \qquad (\beta > 0), \tag{A.12}$$

which has the global minimum $f_* = 0$ at $\boldsymbol{x}_* = (1, 2, ..., n)$ in the search domain $-n \le x_i \le n$ for $i = 1, ..., n$. A related function

$$f(\boldsymbol{x}) = \sum_{j=1}^{n} \Big\{ \sum_{i=1}^{n} (i + \beta)\Big[x_i^j - (\frac{1}{i})^j\Big]\Big\}^2, \tag{A.13}$$

has the global minimum $f_* = 0$ at $(1, 1/2, 1/3, ..., 1/n)$ within the bounds $-1 \le x_i \le 1$ for all $i = 1, 2, ..., n$. As $\beta > 0$ becomes smaller, the global minimum becomes almost indistinguishable from their local minima. In fact, in the extreme case $\beta = 0$, every solution is also a global minimum.

Rastrigin's function:

$$f(\boldsymbol{x}) = 10n + \sum_{i=1}^{n} \Big[x_i^2 - 10\cos(2\pi x_i)\Big], \qquad -5.12 \le x_i \le 5.12, \tag{A.14}$$

whose global minimum is $f_* = 0$ at $(0, 0, ..., 0)$. This function is highly multimodal.

Rosenbrock's function:

$$f(\boldsymbol{x}) = \sum_{i=1}^{n-1} \Big[(x_i - 1)^2 + 100(x_{i+1} - x_i^2)^2\Big], \tag{A.15}$$

whose global minimum $f_* = 0$ occurs at $\boldsymbol{x}_* = (1, 1, ..., 1)$ in the domain $-5 \le x_i \le 5$ where $i = 1, 2, ..., n$. In the 2D case, it is often written as

$$f(x, y) = (x - 1)^2 + 100(y - x^2)^2, \tag{A.16}$$

which is often referred to as the banana function.

Schwefel's function:

$$f(\boldsymbol{x}) = -\sum_{i=1}^{n} x_i \sin\Big(\sqrt{|x_i|}\Big), \qquad -500 \le x_i \le 500, \tag{A.17}$$

whose global minimum $f_* \approx -418.9829n$ occurs at $x_i = 420.9687$ where $i = 1, 2, ..., n$.

Six-hump camel back function:

$$f(x, y) = (4 - 2.1x^2 + \frac{1}{3}x^4)x^2 + xy + 4(y^2 - 1)y^2, \tag{A.18}$$

where $-3 \le x \le 3$ and $-2 \le y \le 2$. This function has two global minima $f_* \approx -1.0316$ at $(x_*, y_*) = (0.0898, -0.7126)$ and $(-0.0898, 0.7126)$.

Shubert's function:

$$f(\boldsymbol{x}) = \Big[\sum_{i=1}^{n} i \cos\Big(i + (i+1)x\Big)\Big] \cdot \Big[\sum_{i=1}^{n} i \cos\Big(i + (i+1)y\Big)\Big], \tag{A.19}$$

which has 18 global minima $f_* \approx -186.7309$ for $n = 5$ in the search domain $-10 \le x, y \le 10$.

Xin-She Yang's functions:

$$f(\boldsymbol{x}) = \Big(\sum_{i=1}^{n} |x_i|\Big) \exp\Big[-\sum_{i=1}^{n} \sin(x_i^2)\Big], \tag{A.20}$$

which has the global minimum $f_* = 0$ at $\boldsymbol{x}_* = (0, 0, ..., 0)$ in the domain $-2\pi \le x_i \le 2\pi$ where $i = 1, 2, ..., n$. This function is not smooth, and its derivatives are not well defined at the optimum $(0, 0, ..., 0)$.

A related function is

$$f(\boldsymbol{x}) = -\Big(\sum_{i=1}^{n} |x_i|\Big) \exp\Big(-\sum_{i=1}^{n} x_i^2\Big), \qquad -10 \le x_i \le 10, \tag{A.21}$$

which has multiple global minima. For example, for $n = 2$, we have 4 equal minima $f_* = -1/\sqrt{e} \approx -0.6065$ at $(1/2, 1/2)$, $(1/2, -1/2)$, $(-1/2, 1/2)$ and $(-1/2, -1/2)$.

Yang also designed a standing-wave function with a defect

$$f(\boldsymbol{x}) = \left[e^{-\sum_{i=1}^{n}(x_i/\beta)^{2m}} - 2e^{-\sum_{i=1}^{n} x_i^2}\right] \cdot \prod_{i=1}^{n} \cos^2 x_i, \qquad m = 5, \tag{A.22}$$

which has many local minima and the unique global minimum $f_* = -1$ at $\boldsymbol{x}_* = (0, 0, ..., 0)$ for $\beta = 15$ within the domain $-20 \le x_i \le 20$ for $i = 1, 2, ..., n$. He also proposed another multimodal function

$$f(\boldsymbol{x}) = \left\{[\sum_{i=1}^{n} \sin^2(x_i)] - \exp(-\sum_{i=1}^{n} x_i^2)\right\} \cdot \exp\left[-\sum_{i=1}^{n} \sin^2 \sqrt{|x_i|}\right], \tag{A.23}$$

whose global minimum $f_* = -1$ occurs at $\boldsymbol{x}_* = (0, 0, ..., 0)$ in the domain $-10 \le x_i \le 10$ where $i = 1, 2, ..., n$. In the 2D case, its landscape looks like a wonderful candlestick.

Most test functions are deterministic. Yang designed a test function with stochastic components

$$f(x, y) = -5e^{-\beta[(x-\pi)^2+(y-\pi)^2]} - \sum_{j=1}^{K}\sum_{i=1}^{K} \epsilon_{ij} e^{-\alpha[(x-i)^2+(y-j)^2]}, \tag{A.24}$$

where $\alpha, \beta > 0$ are scaling parameters, which can often be taken as $\alpha = \beta = 1$. Here ϵ_{ij} are random variables and can be drawn from a uniform distribution $\epsilon_{ij} \sim$ Unif[0,1]. The domain is $0 \le x, y \le K$ and $K = 10$. This function has K^2 local valleys at grid locations and the fixed global minimum at $\boldsymbol{x}_* = (\pi, \pi)$. It is worth pointing that the minimum $f_{\min}$ is random, rather than a fixed value; it may vary from $-(K^2 + 5)$ to -5, depending α and β as well as the random numbers drawn.

Furthermore, he also designed a stochastic function

$$f(\boldsymbol{x}) = \sum_{i=1}^{n} \epsilon_i \left| x_i - \frac{1}{i} \right|, \qquad -5 \le x_i \le 5, \tag{A.25}$$

where ϵ_i $(i = 1, 2, ..., n)$ are random variables which are uniformly distributed in $[0, 1]$. That is, $\epsilon_i \sim$Unif$[0, 1]$. This function has the unique minimum $f_* = 0$ at $\boldsymbol{x}_* = (1, 1/2, ..., 1/n)$ which is also singular.

Zakharov's functions:

$$f(\boldsymbol{x}) = \sum_{i=1}^{n} x_i^2 + \left(\frac{1}{2}\sum_{i=1}^{n} i x_i\right)^2 + \left(\frac{1}{2}\sum_{i=1}^{n} i x_i\right)^4, \tag{A.26}$$

whose global minimum $f_* = 0$ occurs at $\boldsymbol{x}_* = (0, 0, ..., 0)$. Obviously, we can generalize this function as

$$f(\boldsymbol{x}) = \sum_{i=1}^{n} x_i^2 + \sum_{k=1}^{K} J_n^{2k}, \tag{A.27}$$

where $K = 1, 2, ..., 20$ and

$$J_n = \frac{1}{2} \sum_{i=1}^{n} i x_i. \tag{A.28}$$

REFERENCES

1. D. H. Ackley, *A Connectionist Machine for Genetic Hillclimbing*, Kluwer Academic Publishers, 1987.
2. C. A. Floudas, P. M., Pardalos, C. S. Adjiman, W. R. Esposito, Z. H. Gumus, S. T. Harding, J. L. Klepeis, C. A., Meyer, C. A. Scheiger, *Handbook of Test Problems in Local and Global Optimization*, Springer, 1999.
3. A. Hedar, Test function web pages, http://www-optima.amp.i.kyoto-u.ac.jp /member/student/hedar/Hedar_files/TestGO_files/Page364.htm
4. M. Molga, C. Smutnicki, "Test functions for optimization needs", http://www.zsd.ict.pwr.wroc.pl/files/docs/functions.pdf
5. X. S. Yang, "Firefly algorithm, Lévy flights and global optimization", in: *Research and Development in Intelligent Systems XXVI*, (Eds M. Bramer *et al.*), Springer, London, pp. 209-218 (2010).
6. X. S. Yang and S. Deb, "Engineering optimization by cuckoo search", *Int. J. Math. Modeling and Numerical Optimization*, **1**, No. 4, (in press) (2010).
7. X. S. Yang, "Firefly algorithm, stochastic test functions and design optimization", *Int. J. Bio-inspired Computation*, **2**, No. 2, 78-84 (2010).