

ENGINEERING OPTIMIZATION





ENGINEERING OPTIMIZATION

**An Introduction With Metaheuristic
Applications**

Xin-She Yang

University of Cambridge, United Kingdom

A JOHN WILEY & SONS, INC., PUBLICATION



Copyright ©2010 by John Wiley & Sons, Inc. All rights reserved.

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

No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, scanning, or otherwise, except as permitted under Section 107 or 108 of the 1976 United States Copyright Act, without either the prior written permission of the Publisher, or authorization through payment of the appropriate per-copy fee to the Copyright Clearance Center, Inc., 222 Rosewood Drive, Danvers, MA 01923, (978) 750-8400, fax (978) 646-8600, or on the web at www.copyright.com. Requests to the Publisher for permission should be addressed to the Permissions Department, John Wiley & Sons, Inc., 111 River Street, Hoboken, NJ 07030, (201) 748-6011, fax (201) 748-6008.

Limit of Liability/Disclaimer of Warranty: While the publisher and author have used their best efforts in preparing this book, they make no representations or warranties with respect to the accuracy or completeness of the contents of this book and specifically disclaim any implied warranties of merchantability or fitness for a particular purpose. No warranty may be created or extended by sales representatives or written sales materials. The advice and strategies contained herein may not be suitable for your situation. You should consult with a professional where appropriate. Neither the publisher nor author shall be liable for any loss of profit or any other commercial damages, including but not limited to special, incidental, consequential, or other damages.

For general information on our other products and services please contact our Customer Care Department with the U.S. at 877-762-2974, outside the U.S. at 317-572-3993 or fax 317-572-4002.

Wiley also publishes its books in a variety of electronic formats. Some content that appears in print, however, may not be available in electronic format.

Library of Congress Cataloging-in-Publication Data:

Engineering Optimization: An Introduction with Metaheuristic Applications / Xin-She Yang
"Wiley-Interscience."
Includes bibliographical references and index.
ISBN 978-0-470-58246-6
1. ?? 2. ??

Printed in the United States of America.

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



LIST OF FIGURES

1.1	Reflection of light at a mirror.	11
2.1	Classification of optimization problems.	18
2.2	Classification of algorithms.	21
3.1	(a) A jump discontinuity at x_0 , but piecewise continuous where the solid point means the point is included, while a circle is excluded. (b) $ \sin(x) $ is piecewise smooth (but not differentiable at $x = 0, \pm\pi, \pm2\pi, \dots$).	35
3.2	(a) $ x $ is not differentiable at $x = 0$, (b) $1/x$ has a singular point at $x = 0$.	36
3.3	The sign of the second derivative at a stationary point. (a) $f''(x) > 0$, (b) $f''(x) = 0$, and (c) $f''(x) < 0$.	37
3.4	Sine function $\sin(x)$ and its stationary points (marked with $-$) and points of inflection (marked with \circ).	38

3.5	Strong and weak maxima and minima. A is a weak local maximum; point B is a local maximum with discontinuity; C and D are the minimum and maximum, respectively. E is a weak local minimum. Point F corresponds to a strong maximum and also the global maximum, while point G is the strong global minimum.	39
3.6	Convexity: (a) non-convex, and (b) convex.	54
3.7	Convexity: (a) affine set $\mathbf{x} = \theta\mathbf{x}_1 + (1 - \theta)\mathbf{x}_2$ where $\theta \in \mathfrak{R}$, (b) convex hull $\mathbf{x} = \sum_{i=1}^k \theta_i \mathbf{x}_i$ with $\sum_{i=1}^k \theta_i = 1$ and $\theta_i \geq 0$, and (c) convex cone $\mathbf{x} = \theta_1 \mathbf{x}_1 + \theta_2 \mathbf{x}_2$ with $\theta_1 \geq 0$ and $\theta_2 \geq 0$.	54
3.8	Convexity of a function $f(x)$. Chord AB lies above the curve segment joining A and B . For any point P , we have $L_\alpha = \alpha L$, $L_\beta = \beta L$ and $L = x_B - x_A $.	55
4.1	The basic steps of a line search method.	66
4.2	Schematic representation of linear programming. If $\alpha = 2$, $\beta = 3$, $n_1 = 16$, $n_2 = 10$ and $n = 20$, then the optimal solution is at $B(10, 10)$.	69
4.3	Minimization of a function with the two equality constraints.	79
4.4	The feasible region, infeasible solutions (marked with \circ) and the optimal point (marked with \bullet).	81
5.1	The pseudocode of the BFGS method.	86
5.2	The concept of a simplex: (a) 1-simplex, (b) 2-simplex, and (c) 3-simplex.	87
5.3	Simplex manipulations: (a) reflection with fixed volume (area), (b) expansion or contraction along the line of reflection, (c) reduction.	87
5.4	Pseudocode of Nelder-Mead's downhill simplex method.	89
5.5	Pseudocode of a trust region method.	90
5.6	Procedure of sequential quadratic programming.	93
6.1	Newton's method for the equality constrained optimization.	100
6.2	The barrier method: (a) log barrier near a boundary, and (b) central path for $n = 2$ and $N = 4$.	102
6.3	The procedure of the barrier method for convex optimization.	104

6.4	Robustness of the optimal solution.	106
7.1	Concept of curvature.	112
7.2	The curvature of a circle at any point is $1/r$.	113
7.3	Variations in the path $y(x)$.	114
7.4	Geodesic path on the surface of a sphere.	116
7.5	A simple pendulum.	119
8.1	Histogram of 5000 random numbers generated from a uniform distribution in the range $(0, 1)$.	135
8.2	Histogram of the normally-distributed numbers generated by the simple inverse transform method.	139
9.1	Estimating π by repeatedly dropping needles or tossing coins.	144
9.2	Representation of Monte Carlo integration.	147
9.3	Pseudo code for Monte Carlo integration.	148
10.1	Random walk in a one-dimensional line. At any point, the probability moving to the left or right equals to $1/2$.	156
10.2	Random walk and the path of 100 consecutive steps starting at position 0.	157
10.3	Brownian motion in 2D: random walk with a Gaussian step-size distribution and the path of 100 steps starting at the origin $(0, 0)$ (marked with \bullet).	158
10.4	Lévy flights in 2D setting starting at the origin $(0, 0)$ (marked with \bullet).	160
10.5	Metropolis-Hastings algorithm.	165
10.6	The Ghate-Smith Markov chain algorithm for optimization.	168
11.1	Pseudo code of genetic algorithms.	175
11.2	Diagram of crossover at a random crossover point (location) in genetic algorithms.	175
11.3	Schematic representation of mutation at a single site by flipping a randomly selected bit ($1 \rightarrow 0$).	176
11.4	Encode all design variables into a single long string.	178
11.5	Easom's function: $f(x) = -\cos(x)e^{-(x-\pi)^2}$ for $x \in [-10, 10]$ has a unique global maximum $f_{\max} = 1$ at $x_* = \pi$.	178

11.6	Typical outputs from a typical run. The best estimate will approach π while the fitness will approach $f_{\max} = 1$.	179
12.1	Simulated annealing algorithm.	183
12.2	Rosenbrock's function with the global minimum $f_* = 0$ at $(1, 1)$.	185
12.3	500 evaluations during the simulated annealing. The final global best is marked with \bullet .	185
12.4	The egg crate function with a global minimum $f_* = 0$ at $(0, 0)$.	186
12.5	The paths of moves of simulated annealing during iterations.	187
13.1	Pseudo code of ant colony optimization.	191
13.2	The double bridge problem for routing performance: route (2) is shorter than route (1).	192
13.3	Route selection via ACO: (a) initially, ants choose each route with a 50-50 probability, and (b) almost all ants move along the shorter route after 5 iterations.	193
13.4	Landscape and pheromone distribution of the multi-peak function.	194
14.1	Pseudo code of bee algorithms	199
15.1	Schematic representation of the motion of a particle in PSO, moving towards the global best \mathbf{g}^* and the current best \mathbf{x}_i^* for each particle i .	204
15.2	Pseudo code of particle swarm optimization.	205
15.3	A multimodal function with the global minimum $f_* = 0$ at $(0, 0)$, however, it has a singularity at $(0, 0)$ (right).	207
15.4	Michalewicz function with a global minimum at about $(2.20319, 1.57049)$.	208
15.5	Initial locations and final locations of 20 particles after 10 iterations.	209
16.1	Harmony of two notes with a frequency ratio of 2:3 and their waveform.	214
16.2	Random music notes.	215
16.3	Pseudo code of Harmony Search.	216
16.4	The variations of harmonies in harmony search.	217

16.5	Yang's standing wave function with the global minimum at $(0, 0)$.	218
17.1	Pseudo code of the firefly algorithm (FA).	223
17.2	Landscape of a function with two equal global maxima.	226
17.3	The initial locations of 25 fireflies (left) and their final locations after 20 iterations (right).	227
17.4	Landscape of Ackley's 2D function with the global minimum 0 at $(0, 0)$.	228
17.5	The initial locations of the 25 fireflies (left) and their final locations after 20 iterations (right).	229
18.1	Non-dominated set, Pareto front and ideal vectors in a minimization problem with two objectives f_1 and f_2 .	236
18.2	Three functions reach the global minimum at $x_* = \beta, y_* = \alpha - \gamma$.	238
18.3	Final locations of 40 particles after 5 iterations. The optimal point is at $(1/3, 0)$ marked with \circ .	239
18.4	Finding the Pareto solution with maximum utility in a maximization problem with two objectives.	241
18.5	Pareto front is the line connecting $A(5, 0)$ and $B(0, 5/\alpha)$. The Pareto solution with maximum utility is $U_* = 25$ at point A.	242
19.1	The design optimization of a simple spring.	248
19.2	Pressure vessel design and optimization.	249
19.3	The rectangular design domain is divided into N elements. As optimization and material distribution evolve, the shape becomes a truss-style structure (bottom).	250
19.4	Harmonic vibrations.	254
19.5	A rectangular beam with inhomogeneous materials properties (in 10 different cells).	257
D.1	Heron's proof of the shortest path.	307
D.2	The feasible region of design variables of a simple linear programming problem.	308
D.3	The plot of $\text{sinc}(x) = \sin(x)/x$.	309
D.4	The plot of $f(x) = x^2 + 25 \cos^2(x)$.	309
D.5	Quadratic penalty function $\Pi(x, \mu) = 100(x - 1)^2 + \pi + \frac{\mu}{2}(x - a)^2$ and $\mu = 2000$.	311
D.6	A simple route to tour 4 cities.	312



PREFACE

Optimization is everywhere, from engineering design to computer sciences and from scheduling to economics. However, to realize that everything is optimization does not make the problem-solving easier. In fact, many seemingly simple problems are very difficult to solve. A well-known example is the so-called Traveling Salesman Problem in which the salesman intends to visit, say, 50 cities, exactly once so as to minimize the overall distance traveled or the overall traveling cost. No efficient algorithms exist for such hard problems. The latest developments over the last two decades tend to use metaheuristic algorithms. In fact, a vast majority of modern optimization techniques are usually heuristic and/or metaheuristic. Metaheuristic algorithms such as Simulated Annealing, Particle Swarm Optimization, Harmony Search, and Genetic Algorithms are becoming very powerful in solving hard optimization problems, and they have been applied in almost all major areas of science and engineering as well as industrial applications.

This book introduces all the major metaheuristic algorithms and their applications in optimization. This textbook consists of three parts: Part I: Introduction and fundamentals of optimization and algorithms; Part II: Metaheuristic algorithms; and Part III: applications of metaheuristics in engineering optimization. Part I provides a brief introduction to the nature of optimization and the common approaches to optimization problems, random

number generation and Monte Carlo simulations. In Part II, we introduce all major/widely used metaheuristic algorithms in great detail, including Genetic Algorithms, Simulated Annealing, Ant Algorithms, Bee Algorithms, Particle Swarm Optimization, Firefly Algorithms, Harmony Search and others. In Part III, we briefly introduce multi-objective optimization. We also discuss a wide range of applications using metaheuristic algorithms in solving real-world optimization problems. In the appendices, we provide the implementation of some of the important/popular algorithms in Matlab[®] and/or Octave so that readers can use them for learning or solving other optimization problems. The files of the computer programs in the book are available at Wiley's FTP site ftp://ftp.wiley.com/public/sci_tech_med/engineering_optimization

This unique book is self-contained with many step-by-step worked examples including various exercises. It can serve as an ideal textbook for both students and researchers to learn modern metaheuristic algorithms and engineering optimization.

XIN-SHE YANG

*Cambridge, UK
April, 2010*

ACKNOWLEDGMENTS

I would like to thank many of my mentors, friends, and colleagues for their help: J. Brindley, A. C. Fowler, A. B. Forbes, C. J. McDiarmid, A. C. McIntosh, G. T. Parks, S. Tsou, and L. Wright. Special thanks to my students at Cambridge University: E. Flower, M. Jordan, C. Pearson, J. Perry, P. De Souza, M. Stewart, and H. Scott Whittle.

I also would like to thank my Editor, Susanne Steitz-Filler, Editorial Program Coordinator, Jacqueline Palmieri, Production Editor, Melissa Yanuzzi, Copyeditor, Sharon Short, and staff at Wiley for their help and professionalism.

Last but not least, I thank my wife and son for their support and help.

X. S. Y.



INTRODUCTION

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

$$\underset{\mathbf{x} \in \mathbb{R}^n}{\text{minimize}} \quad f_i(\mathbf{x}), \quad (i = 1, 2, \dots, M), \quad (\text{I.1})$$

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

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

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

$$\mathbf{x} = (x_1, x_2, \dots, x_n)^T, \quad (\text{I.4})$$

where the components x_i of \mathbf{x} are called design or decision variables, and they can be real continuous, discrete or a mixture of these two. The functions $f_i(\mathbf{x})$ where $i = 1, 2, \dots, M$ 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 \mathbb{R}^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 $f(\mathbf{x})$ is equivalent to the minimization of $-f(\mathbf{x})$, and any inequality $g(\mathbf{x}) \leq 0$ is equivalent to $-g(\mathbf{x}) \geq 0$. 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 $f(\mathbf{x})$. 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

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

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

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

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

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

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

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

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

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 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 petaflops; 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 understanding 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 optimization", *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).