# ENGINEERING OPTIMIZATION

# ENGINEERING OPTIMIZATION

## An Introduction With Metaheuristic Applications

**Xin-She Yang**
University of Cambridge, United Kingdom

**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:**

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

# 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

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

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

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

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 $\phi_j$ and inequalities for $\psi_k$ are called constraints. It is worth pointing out that we can also write the inequalities in the other way $\geq 0$, 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 $\phi_j$ and $\psi_k$ 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 $f_i, \phi_j$ and $\psi_k$ 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 $3 \times 10^{15}$ 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).